# Snippets

Al Mehedi / Server Configuration

Stop watching Clone Edit Delete

Source Revisions

Created by Al Mehedi last modified 2 minutes ago

server_configuration.markdown Raw

# Initial OS Updates

## CentOS 8

**CentOS Linux distribution is a stable, predictable, manageable and reproducible platform derived from the sources of Red Hat Enterprise Linux (RHEL). CentOS 8 support is guaranteed until 31 May 2029.**

### System Updates `CentOS 8`

```
# dnf check-update
# dnf update
# dnf distro-sync
# dnf autoremove
```

# Installing NTP

The Network Time Protocol (NTP) is a protocol used to synchronize computer system clock automatically over a networks. The machine can have the system clock use Coordinated Universal Time (UTC) rather than local time. NTP server is important component in every enterprise system since it gets the network time synchronized, scripts execute on time, backup jobs done and more.

## CentOS 8

On CentOS 8 ntp is implemented by chronyd. Chrony works both as an NTP server and as an NTP client, which is used to synchronize the system clock with NTP servers, and can be used to synchronize the system clock with a reference clock (e.g a GPS receiver).

Check if `chrony` is installed on your system:

```
# dnf list installed chrony
```

If not installed then install `chrony` by running:

```
# dnf install chrony
```

Now start the chronyd service, enable it to auto start at system boot and verify the running status using the following:

```
# systemctl start chronyd
# systemctl enable chronyd
# systemctl status chronyd
```

Now run the following command to show the current time sources (NTP server) that chronyd is accessing.

```
# chronyc sources
```

Set NTP synchronization by entering:

```
# timedatectl set-ntp true
# timedatectl status
```

To Install NTPStat, it's possible to display time synchronization status.

```
# dnf install ntpstat
# ntpstat
```

### Configure NTP Server Using Chrony [Optional]

To set up your CentOS 8 server as a master NTP time server. Open the `/etc/chrony.conf` configuration file

```
# nano /etc/chrony.conf
```

Then look for the `allow` configuration directive and uncomment it and set its value to the *network* or *subnet* address from which the clients are allowed to connect (e.g.).

```
allow 192.168.56.0/24
```

Restart the `chronyd` via `systemctl`. Next, open access to the NTP service in firewalld configuration to allows for incoming NTP requests from clients.

```
# firewall-cmd --permanent --add-service=ntp
# firewall-cmd --reload
```

On the server, run the following command to display information about NTP clients assessing the NTP server.

```
# chronyc clients
```

For more information on how to use the chronyc utility, run the following command.

```
# man chronyc
```

## Installing Build Essentials / Development Tools

### CentOS 8

On RHEL 8 / CentOS 8 The `development tools` group acts as a transitional package for installation of multiple development, compilation and debugging tools. Most notably these include Automake, Autoconf, Gcc (C/C++) as well as various Perl & Python macros and debuggers.

Execute the following command to see the list of all packages available within the Development Tools group:

```
# dnf groupinfo "Development Tools"
```

Execute the following command to install Development Tools:

```
# dnf group install "Development Tools"
```

#### Installing EPEL

The EPEL repository is an additional package repository that provides easy access to install packages for commonly used software.

```
# dnf search epel
# dnf info epel-release
# dnf install epel-release
```

Enable the PowerTools repository since EPEL packages may depend on packages from it:

```
# dnf config-manager --set-enabled PowerTools
```

Now update the package manager to list your new repos by running:

```
# dnf update
# dnf repolist
```

## Editing Hosts and Hostname File

The hostname is a translation of your IP address, it is your device ID in the network, like a domain name (a hostname can be a domain name). Hostname's function is to make the address easy for humans so we don't need to remember IP addresses. Two devices can't have share a hostname, it is unique.

The device's hostname in Linux is stored in two configuration files located in the directory `/etc/`. The files `/etc/hostname` and `/etc/hosts`.

To verify your current hostname run:

```
# hostname
```

`Systemd` based Linux distro such as Ubuntu Linux 16.04 LTS and above can simply use the `hostnamectl` command to change hostname. To set and see current setting just type the following command:

```
# hostnamectl set-hostname <NewHostname>
# hostnamectl
```

The file `/etc/hosts` maps the host and ip addresses, it translates from IP addresses (X.X.X.X) to FQDN e.g. domain.com or @Anything. In this file you can add all the local network devices ip addresses and `hostnames` to avoid domain name resolution. Usually the file hosts has 3 columns, the IP address, the hostname and an alias for the hostname, it can have more than one alias or may not have at all as in the example below.

```
127.0.0.1               localhost
<InstanceIPAddress>     example.com                  <NewHostname>

# The following lines are desirable for IPv6 capable hosts
::1     localhost ip6-localhost ip6-loopback
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
```

*On the first line `localhost` is preferred by services that are relies on loopback address, you can add any name and translate to desired ip address, keep in mind that `<DropletIPAddress>` only be translated from alias to FQDN to ip address if you have set the FQDN as the RDNS on your instance*

The `/etc/hosts` file act as a local DNS. We can check if the `FQDN` and `<NewHostname>` pointing to the right `<InstanceIPAddress>` by running:

```
# ping example.com
# ping <NewHostname>
```

Reboot the system to changes take effect:

```
# reboot
```

## Managing New User

Since Linux is a multi-user operating system, several people may be logged in and actively working on a given machine at the same time. Security-wise, it is never a good idea to allow users to share the credentials of the same account. In fact, best practices dictate the use of as many user accounts as people needing access to the machine. At the same time, it is to be expected that two or more users may need to share access to certain system resources, such as directories and files. User and group management in Linux allows us to accomplish both objectives.

Local accounts or users in Linux like operating system is managed by `useradd`, `usermod`, `userdel`, `chage` and `passwd` commands.

- `adduser` : add a user to the system.
- `userdel` : delete a user account and related files.
- `addgroup` : add a group to the system.
- `delgroup` : remove a group from the system.
- `usermod` : modify a user account.
- `chage` : change user password expiry information.
- `sudo` run one or more commands as another user (typically with superuser permissions).

*Relevant files: `/etc/passwd` (user information), `/etc/shadow` (encrypted passwords), `/etc/group` (group information) and `/etc/sudoers` (configuration for sudo).*

### Adding a New Regular Account

```
# adduser <account-name>
```

You may be prompted to set the new user's initial password, and other optional information (such as full name, work phone, etc). This will be stored in `/etc/passwd` using colons as field separators. If not, you can assign a password for the newly created account named `<account-name>` with:

```
# passwd <account-name>
```

*When a new user is added, a group with the same name is created automatically. This is called a primary group*

### The `/etc/sudoers` File

To grant `<account-name>` superuser permissions, we will need to add an entry for it in `/etc/sudoers`. This file is used to indicate which users can run what commands with elevated permissions (most likely as root).

Although `/etc/sudoers` is nothing more and nothing less than a plain text file, it **must NOT be edited using a regular text editor**. Instead, we will use the `visudo` command. As opposed to other text editors, by utilizing `visudo` we will ensure that

- No one else can modify the file at the same time.
- The file syntax is checked upon saving changes.

The easiest method to grant superuser permissions for `<account-name>` is by adding the following line at the bottom of `/etc/sudoers`:

```
## Allows <account_name> to run any commands anywhere
<account_name>       ALL=(ALL)       ALL
```

Let's explain the syntax of this line:

- First off, you indicate which user this rule refers to ( `<account-name>` ).
- The first ALL means the rule applies to all hosts using the same `/etc/sudoers` file. Nowadays, this means the current host since the same file is not shared across other machines.

- Next, (ALL) ALL tells us that `<account-name>` will be allowed to run all commands as any user. Functionally speaking, this is equivalent to (root) ALL.

Now run `visudo` and press enter. `visudo` will autometically open the `/etc/sudoers` file. After editing the file press `esc` and type `:wq` to save and press `enter`.

To check the user role in the system run the following command and it will print `root` if the user has sudo privileged.

```
$ sudo whoami
```

## Customizing Bash Prompt

In BASH, we can customize and change the BASH prompt as the way you want by changing the value of `PS1` environment variable.

Customize the **root account** prompt by editing `~/.bashrc` as root and paste the following lines at the end of the file:

```
#Custom color for PS1.

CYAN="\e[0;36m"
BLUE="\e[0;34m"
WHITE="\e[00m"
CEND="\e[m"

RED_BACKGROUND="\e[41m"
GREEN_BACKGROUND="\e[42m"

PS1="\[$WHITE\]\[$RED_BACKGROUND\]\u\[$WHITE\]@\[$CYAN\]\h: \[$BLUE\]\w\[$CEND\]>> "
```

Customize the **user account** prompt by editing `~/.bashrc` as user and paste the following lines at the end of the file:

```
#Custom color for PS1.

CYAN="\e[0;36m"
BLUE="\e[0;34m"
WHITE="\e[00m"
CEND="\e[m"

LIGHT_RED="\e[91m"
LIGHT_GREEN="\e[92m"

PS1="\[$WHITE\]\[$LIGHT_GREEN\]\u\[$WHITE\]@\[$CYAN\]\h: \[$BLUE\]\w\[$CEND\]>> "
```

## Editing SSH Login Banner

A legal banner contains some security warning information or general information, that alerts the user. It can be used for security, legal info, company policy, etc. To display Welcome or Warning message for SSH users before login. Use `issue.net` file to display a banner massages.

`/etc/issue.net` and `/etc/issue` are used to display a banner. `/etc/issue.net` is shown to the users who connect from the network. `/etc/issue` is shown to both local users and network users unless `/etc/issue.net` is present and configured.

Open the following file with the editor:

```
# nano /etc/issue.net
```

Copy and paste the following lines

```
careful_what_you_wish_for__you_may_receive_it!
```

To configure them to be displayed when you login via SSH, you need to uncomment `#Banner` and specify the desired filename at `/etc/ssh/sshd_config`, like:

```
Banner /etc/issue.net
```

## Configuring SSH Daemon

Edit the SSH daemon `/etc/ssh/sshd_config` configuration file and paste following lines at the end of the file.

```
# This is the sshd server system-wide configuration file.  See
# sshd_config(5) for more information.

# This sshd_config was a successor of systems sshd_config.default

# The strategy used for options in the default sshd_config shipped with
# OpenSSH is to specify options with their default value where
# possible, but leave them commented.  Uncommented options change a
```

```
# default value.

# Login Banner
Banner /etc/issue.net
PrintMotd yes

# Allow Access
AllowUsers <user-account>

# Listening Ports, IPs and protocols
Port 22

# Use these options to restrict which interfaces/protocols sshd will bind to
#ListenAddress ::
#ListenAddress 0.0.0.0
AddressFamily inet
Protocol 2

# HostKeys for protocol version 2
HostKey /etc/ssh/ssh_host_rsa_key
HostKey /etc/ssh/ssh_host_ecdsa_key
HostKey /etc/ssh/ssh_host_ed25519_key

# Logging
PrintLastLog yes
SyslogFacility AUTH
LogLevel INFO

# Authentication:
LoginGraceTime 120
MaxAuthTries 5
MaxSessions 5
PermitRootLogin no
StrictModes yes

# TCP connection
TCPKeepAlive yes
ClientAliveInterval 15
ClientAliveCountMax 55

# Don't read the user's ~/.rhosts and ~/.shosts files
IgnoreRhosts yes

# similar for protocol version 2
HostbasedAuthentication no

# Uncomment if you don't trust ~/.ssh/known_hosts for RhostsRSAAuthentication
IgnoreUserKnownHosts yes

# To enable empty passwords, change to yes (NOT RECOMMENDED)
PermitEmptyPasswords no

# Change to yes to enable challenge-response passwords (beware issues with
# some PAM modules and threads)
ChallengeResponseAuthentication yes
UsePAM yes

# Change to no to disable tunnelled clear text passwords
PasswordAuthentication no

# override default of no subsystems
#Subsystem  sftp    /usr/lib/misc/sftp-server
Subsystem   sftp    internal-sftp
```

Now test the edited file by running:

```
# sshd -t
```

If all goes well, restart the `sshd` to changes take effect:

```
# systemctl restart sshd
```

# Configuring Firewall

In computing, a firewall is a network security system that monitors and controls incoming and outgoing network traffic based on predetermined security rules. A firewall typically establishes a barrier between a trusted internal network and untrusted external network, such as the Internet.

## CentOS 8

`firewalld` is firewall management software available for many Linux distributions, which acts as a frontend for Linux's in-kernel nftables or iptables packet filtering systems. Learn more about firewalld

Verify `firewalld` is running and reachable by running:

```
# firewall-cmd --state
```

We can see which zone is currently selected as the default and if it's active zone by running:

```
# firewall-cmd --get-default-zone
# firewall-cmd --get-active-zones
```

### Adding a Service to your Zones

The most straighforward method is to add the services or ports you need to the zones you are using. You can get a list of the available service definitions with the `--get-services` option:

```
# firewall-cmd --get-services
```

**Note:** *You can get more details about each of these services by looking at their associated `.xml` file within the `/usr/lib/firewalld/services` directory. For instance, the SSH service is defined like this: `#cat /usr/lib/firewalld/services/ssh.xml`*

You can enable a service for a zone using the `--add-service=` parameter. The operation will target the default zone or whatever zone is specified by the `--zone=` parameter. By default, this will only adjust the current firewall session. You can adjust the permanent firewall configuration by including the `--permanent` flag.

Let's add some common services by running:

```
# firewall-cmd --zone=public --add-service=ssh --permanent
# firewall-cmd --zone=public --add-service=http --permanent
# firewall-cmd --zone=public --add-service=https --permanent
```

We can verify the operation was successful by using the `--list-all` or `--list-services` operations:

```
# firewall-cmd --zone=public --list-services --permanent
```

*Leave the `--permanent` flag if you only want to see non-permanent running services.*

### Defining a Service If No Appropriate Service Is Available

The services that are included with the firewalld installation represent many of the most common applications that you may wish to allow access to. However, there will likely be scenarios where these services do not fit your requirements.

Using services is easier to administer than ports, but requires a bit of up-front work. The easiest way to start is to copy an existing script (found in `/usr/lib/firewalld/services`) to the `/etc/firewalld/services` directory where the firewall looks for non-standard definitions.

Check the Defining a Service for more details.

Reload your firewall to get access to your new service:

```
# firewall-cmd --reload
```

You can see that it is now among the list of available services:

```
# firewall-cmd --get-services
```

*You can now use this service in your zones as you normally would.*

You can remove a service by running:

```
# firewall-cmd --zone=public --remove-service=<Service Name> --permanent
# firewall-cmd --reload
```

You can see a service info by running:

```
# firewall-cmd --info-service=<Service Name>
```

## Tuning Linux Kernel

Follow the Linux Kernel Tuning to setup an instance for the first time.

*Though `Linux Kernal Tuning` is suitable for small instance, It's not a suitable choice for every situations, Therefore apply the `Special Use Cases` section from the end of `Linux Kernal Tuning` documentation upon system upgrades.*

## Managing SELinux

Linux is regarded as one of the most secure operating systems you can use today, that is because of its illustrious security implementation features such as SELinux (Security-Enhanced Linux).

For starters, SELinux is described as a mandatory access control (MAC) security structure executed in the kernel. SELinux offers a means of enforcing some security policies which would otherwise not be effectively implemented by a System Administrator.

When you install RHEL/CentOS or several derivatives, the SELinux feature or service is enabled by default, due to this some applications on your system may not actually support this security mechanism. Therefore, to make such applications function normally, you have to disable or turn off SELinux.

The first thing to do is to check the status of SELinux on your system, and you can do this by running the following command:

```
# sestatus
```

To make changes on `SELinux` policy, open the file `/etc/sysconfig/selinux` and change the `SELINUX=enforcing` to `SELINUX=permissive`.

```
Reboot the system to changes take effect
```

## Configuring Git

By default git is already installed. Execute the following lines for configuration.

```
# git config --global user.name "Full Name"
# git config --global user.email "Email Address"
# git config --global core.editor nano
# git config --global color.ui true
```

To show all configuration run:

```
# git config --list
```

Configuration for remote repository

```
# git remote add origin "url.git"
# git remote -v
# git pull origin master
# git push origin master
```

In case of error: "fatal: refusing to merge unrelated histories" use `--allow-unrelated-histories` flag. For more details run: `# man git`.

## Managing Certbot

Certbot is a free, open source software tool for automatically using Let's Encrypt certificates on manually-administrated websites to enable HTTPS.

### CentOS 8

Install Certbot by running:

```
# dnf info certbot
# dnf install certbot
```

For additional info `# man certbot`

### Accruing New Certificates

*Make sure to stop any http service on port 80 before running the following command*

```
# certbot certonly --standalone -d domain.com -d www.domain.com
```

You will find the certificates at `/etc/letsencrypt/live` directory. For renewing the certificates run `# certbot renew`

**IMPORTANT: After generating a new certificate or renewing, make sure you copy `chain.pem` file content and paste at the bottom of the `cert.pem` file.**

**NOTE: Rustls library expect `privkey.pem` file to be start with `-----BEGIN RSA PRIVATE KEY-----` and end with `-----END RSA PRIVATE KEY-----`. Make sure to check on the file content in case of error.**

## Rust (Programming language)

Installing Rust in Linux

### Installing Diesel

Diesel is a Safe, Extensible ORM and Query Builder for Rust. Diesel is the most productive way to interact with databases in Rust because of its safe and composable abstractions over queries.

Diesel provides a separate CLI tool to help manage your project. Since it's a standalone binary, and doesn't affect your project's code directly, we don't add it to `Cargo.toml`. Instead, we just install it on our system.

```
# apt update
# apt install libpq-dev
# cargo install diesel_cli --no-default-features --features postgres
```

*`libpq-dev` library is required for diesel to work with PostgreSQL database.*

On `CentOS 8` install `dnf install libpq-devel` package before you compile diese-cli.

For any additional help visit diesel or run:

```
# diesel --help
```

You can also run `diesel SUBCOMMAND -h` to get more information about that sub command.

## Database Configuration

### PostgreSQL

Installing PostgreSQL 12 in Linux

## Daemon Configuration

---

### CentOS 8

Manage process using systemd

**Comments (0)**

What would you like to say?