Rocky Linux Middlewares Guide (Italian version)

A book from the Documentation Team

Version : 2025/07/12

Rocky Documentation Team

Table of contents

1. Server di Database MariaDB

1.1 Prerequisiti

- Un server Rocky Linux
- Saper utilizzare un editor a riga di comando (in questo esempio si utilizza *vi*)
- Un livello di comfort elevato con l'immissione di comandi dalla riga di comando, la visualizzazione dei log e altri compiti generali di amministratore di sistema
- Una comprensione dei database *mariadb-server* è utile
- Eseguire tutti i comandi come root o con *sudo*

1.2 Introduzione

Il *mariadb-server* e il suo client *mariadb* sono le alternative open source a *mysql-server* e *mysql*, e condividono la struttura dei comandi. *mariadb-server* è in esecuzione su molti server web, a causa del popolare Wordpress CMS che lo richiede. Questo database, però, ha molti altri usi.

Se si desidera utilizzare questo insieme ad altri strumenti per il rafforzamento di un server web, consultare la guida Apache Hardened Web Server.

1.3 Installare `mariadb-server`

È necessario installare *mariadb-server*:

```
dnf install mariadb-server
```

1.4 Proteggere `mariadb-server`

Per rafforzare la sicurezza di *mariadb-server* è necessario eseguire uno script, ma prima è necessario abilitare e avviare mariadb:

```
systemctl enable mariadb
```

Quindi:

`systemctl start mariadb`

Poi, eseguite questo comando:

`mysql_secure_installation`

Suggerimento

La versione di mariadb-server che viene abilitata per impostazione predefinita in Rocky Linux 8.5 è la 10.3.32. È possibile installare 10.5.13 abilitando il modulo:

```
dnf module enable mariadb:10.5
```

E poi installare `mariadb`. Dalla versione 10.4.6 di MariaDB, sono disponibili comandi specifici MariaDB che puoi usare al posto dei vecchi comandi `mysql` prefissati. Questi includono il precedentemente menzionato `mysql_secure_installation` che ora può essere chiamato con la versione MariaDB `mariadb-secure-installation`.

Viene visualizzata una finestra di dialogo:

```
NOTE: RUNNING ALL PARTS OF THIS SCRIPT IS RECOMMENDED FOR ALL MariaDB
      SERVERS IN PRODUCTION USE!  PLEASE READ EACH STEP CAREFULLY!

In order to log into MariaDB to secure it, you will need the current
password for the root user.  If you have just installed MariaDB, and
you have not set the root password yet, the password will be blank,
so you should just press enter here.

Enter current password for root (enter for none):
```

Trattandosi di una nuova installazione, non è stata impostata alcuna password di root. Basta premere invio qui.

La parte successiva del dialogo continua:

```
OK, successfully used password, moving on...

Setting the root password ensures that nobody can log into the MariaDB
root user without the proper authorisation.

Set root password? [Y/n]
```

È' assolutamente *necessario* avere una password di root impostata. Ti consigliamo di capire cosa dovrebbe essere e documentarlo in un gestore di password da qualche parte in modo da poterlo estrarre se necessario. Iniziate premendo 'Invio'

per accettare il valore predefinito "Y". Questo farà apparire la finestra di dialogo della password:

```
New password:
Re-enter new password:
```

Inserisci la password scelta e poi confermala inserendola di nuovo. Se questo ha successo, otterrete la seguente finestra di dialogo:

```
Password updated successfully!
Reloading privilege tables..
 ... Success!
```

Il prossimo dialogo riguarda l'utente anonimo:

```
By default, a MariaDB installation has an anonymous user, allowing anyone
to log into MariaDB without having to have a user account created for
them.  This is intended only for testing, and to make the installation
go a bit smoother.  You should remove them before moving into a
production environment.

Remove anonymous users? [Y/n]
```

La risposta qui è "Y", quindi basta premere 'Invio' per accettare l'impostazione predefinita.

La finestra di dialogo procede alla sezione che si occupa di permettere all'utente root di accedere da remoto:

```
... Success!

Normally, root should only be allowed to connect from 'localhost'.  This
ensures that someone cannot guess at the root password from the network.

Disallow root login remotely? [Y/n]
```

root dovrebbe essere necessario solo localmente sulla macchina. Quindi accettate anche questo default premendo 'Invio'.

La finestra di dialogo si sposta poi sul database 'test' che è installato automaticamente con *mariadb-server*:

```
... Success!


By default, MariaDB comes with a database named 'test' that anyone can
access.  This is also intended only for testing, and should be removed
before moving into a production environment.

Remove test database and access to it? [Y/n]
```

Di nuovo, la risposta qui è quella predefinita, quindi basta premere 'Invio' per rimuoverla.

Infine, la finestra di dialogo chiede se si desidera ricaricare i privilegi:

```
- Dropping test database...
 ... Success!
 - Removing privileges on test database...
 ... Success!

Reloading the privilege tables will ensure that all changes made so far
will take effect immediately.

Reload privilege tables now? [Y/n]
```

Anche in questo caso, premere "Invio". Se tutto ha funzionato, si riceverà questo messaggio:

```
 ... Success!

Cleaning up...

All done!  If you've completed all of the above steps, your MariaDB
installation should now be secure.

Thanks for using MariaDB!
```

MariaDB sarà ora pronto all'uso.

1.4.1 Modifiche a Rocky 9.0

Rocky Linux 9.0 utilizza `mariadb-server-10.5.13-2` come versione predefinita di mariadb-server. A partire dalla versione 10.4.3, un nuovo plugin è abilitato automaticamente nel server che cambia la finestra di dialogo `mariadb-secure-installation`. Quel plugin è l'autenticazione `unix-socket`. Questo articolo spiega bene la nuova funzione. Essenzialmente, l'uso dell'autenticazione unix-socket utilizza le credenziali dell'utente connesso per accedere al database. Fa in modo che se l'utente root, ad esempio, accede e quindi utilizza `mysqladmin` per creare o eliminare un database (o qualsiasi altra funzione) non viene richiesta alcuna password per l'accesso. Stesso funzionamento con `mysql`. Ciò significa anche che non c'è una password da compromettere in remoto. Ciò dipende dalla sicurezza degli utenti impostati sul server per la protezione del database.

La seconda finestra di dialogo durante `mariadb-secure-installation` dopo l'impostazione della password per l'utente amministrativo è:

```
Switch to unix_socket authentication Y/n
```

L'impostazione predefinita è "Y", ma anche se si risponde "n", con il plugin abilitato non viene richiesta una password per l'utente, almeno non dall'interfaccia della riga di comando. È possibile specificare password o nessuna password e entrambi funzionano:

```
mysql

MariaDB [(none)]>
```

```
mysql -p
Inserire la password:

MariaDB [(none)]>
```

Per ulteriori informazioni su questa funzione, fare riferimento al link qui sopra. C'è un modo per disattivare questo plugin e tornare ad avere la password come campo obbligatorio, che è anche dettagliato all'interno di tale collegamento.

1.5 Conclusione

Un server di database, come *mariadb-server*, può essere usato per molti scopi. A causa della popolarità del CMS Wordpress, si trova spesso sui server web. Prima di eseguire il database in produzione, tuttavia, è bene rafforzarne la sicurezza.

2. Server DNS privato con `bind`

2.1 Prerequisiti e presupposti

- Un server con Rocky Linux
- Alcuni server interni che necessitano solo di un accesso locale, non tramite Internet
- Diverse postazioni di lavoro che devono accedere agli stessi server presenti sulla stessa rete
- Un buon livello di confidenza con l'inserimento di comandi dalla riga di comando
- Familiarità con un editor a riga di comando (in questo esempio si usa *vi*)
- In grado di utilizzare *firewalld* per la creazione di regole firewall

2.2 Introduzione

I server DNS esterni, o pubblici, mappano gli hostname in indirizzi IP e, nel caso dei record PTR (noti come "pointer" o "reverse"), mappano gli indirizzi IP in hostname. Si tratta di una parte essenziale di Internet. Fa sì che il server di posta, il server web, il server FTP o molti altri server e servizi funzionino come previsto, indipendentemente da dove ci si trovi.

Su una rete privata, in particolare una rete per lo sviluppo di molti sistemi, è possibile utilizzare il file */etc/hosts* della propria workstation Rocky Linux per mappare un nome a un indirizzo IP.

Questo funzionerà per la *vostra* workstation, ma non per qualsiasi altro computer della rete. Il metodo migliore per rendere le cose universalmente applicabili è quello di prendersi un po' di tempo e creare un server DNS locale e privato per gestire questo aspetto per tutti i vostri computer.

Supponiamo di creare server e resolver DNS pubblici a livello di produzione. In questo caso, l'autore raccomanda il più robusto PowerDNS DNS autorevole e ricorsivo, installabile sui server Rocky Linux. Tuttavia, questo documento si riferisce a una rete locale che non espone i propri server DNS al mondo esterno. Ecco perché l'autore ha scelto `bind` per questo esempio.

2.2.1 Spiegazione dei componenti del server DNS

Il DNS separa i servizi in server autoritari e ricorsivi. Attualmente si consiglia di separare questi servizi su hardware o container distinti.

Il server autorevole è l'area di archiviazione di tutti gli indirizzi IP e i nomi host, mentre il server ricorsivo cerca gli indirizzi e i nomi host. Nel caso del nostro server DNS privato, i servizi di server autorevole e ricorsivo lavoreranno insieme.

2.3 Installazione e abilitazione di `bind`

Il primo passo è l'installazione dei pacchetti:

```
dnf install bind bind-utils
```

bind è il demone di servizio di `named`. Abilitare l'avvio al boot:

```
systemctl enable named
```

Avviare `named`:

```
systemctl start named
```

2.4 Configurazione

Prima di apportare modifiche a qualsiasi file di configurazione, creare una copia di backup del file di lavoro originale installato, *named.conf*:

```
cp /etc/named.conf /etc/named.conf.orig
```

Questo aiuterà in futuro se si verificano errori nel file di configurazione. È *sempre* una buona idea fare una copia di backup prima di apportare modifiche.

Modificare il file *named.conf.*. L'autore utilizza *vi*, ma è possibile sostituire l'editor della riga di comando preferito:

```
vi /etc/named.conf
```

Disattivare l'ascolto su localhost. Per farlo, contrassegnate con il segno "#" queste due righe nella sezione "options". In questo modo si interrompe qualsiasi connessione con il mondo esterno.

Questo è utile, soprattutto quando si aggiunge questo DNS alle nostre postazioni di lavoro, perché si vuole che il server DNS risponda solo quando l'indirizzo IP che richiede il servizio è locale e non reagisca se il server o il servizio si trova su Internet.

In questo modo, gli altri server DNS configurati subentreranno quasi immediatamente per cercare i servizi basati su Internet:

```
options {
#       listen-on port 53 { 127.0.0.1; };
#       listen-on-v6 port 53 { ::1; };
```

Infine, si può andare in fondo al file *named.conf* e aggiungere una sezione per la vostra rete. Il nostro esempio è "ourdomain", quindi inserite il nome che volete dare agli host della vostra LAN:

```
# primary forward and reverse zones
//forward zone
zone "ourdomain.lan" IN {
     type master;
     file "ourdomain.lan.db";
     allow-update { none; };
    allow-query {any; };
};
//reverse zone
zone "1.168.192.in-addr.arpa" IN {
     type master;
     file "ourdomain.lan.rev";
     allow-update { none; };
    allow-query { any; };
};
```

Salvare le modifiche (per *vi*, ⇧ Shift + : + W + Q + !)

2.5 I record di forward e reverse

È necessario creare due file in `/var/named`. Questi file verranno modificati se si aggiungono macchine alla rete per includerle nel DNS.

Il primo è il file forward per mappare il nostro indirizzo IP al nome dell'host. Anche in questo caso, il nostro esempio è "ourdomain". Si noti che l'IP del nostro DNS locale è 192.168.1.136. Aggiungere gli host in fondo a questo file.

```
vi /var/named/ourdomain.lan.db
```

Una volta completato, il file avrà un aspetto simile a questo:

```
$TTL 86400
@ IN SOA dns-primary.ourdomain.lan. admin.ourdomain.lan. (
    2019061800 ;Serial
    3600 ;Refresh
    1800 ;Retry
    604800 ;Expire
    86400 ;Minimum TTL
)

;Name Server Information
@ IN NS dns-primary.ourdomain.lan.

;IP for Name Server
dns-primary IN A 192.168.1.136

;A Record for IP address to Hostname
wiki IN A 192.168.1.13
www IN A 192.168.1.14
devel IN A 192.168.1.15
```

Aggiungere tutti gli host e gli indirizzi IP necessari e salvare le modifiche.

È necessario un file reverse per mappare il nostro hostname all'indirizzo IP. In questo caso, l'unica parte dell'IP di cui si ha bisogno è l'ultimo ottetto (in un indirizzo IPv4 ogni numero separato da un "." è un ottetto) dell'host, il PTR e l'hostname.

```
vi /var/named/ourdomain.lan.rev
```

Una volta completato, il file avrà un aspetto simile a questo:

```
$TTL 86400
@ IN SOA dns-primary.ourdomain.lan. admin.ourdomain.lan. (
    2019061800 ;Serial
    3600 ;Refresh
    1800 ;Retry
    604800 ;Expire
    86400 ;Minimum TTL
)
;Name Server Information
@ IN NS dns-primary.ourdomain.lan.

;Reverse lookup for Name Server
136 IN PTR dns-primary.ourdomain.lan.

;PTR Record IP address to HostName
13 IN PTR wiki.ourdomain.lan.
14 IN PTR www.ourdomain.lan.
15 IN PTR devel.ourdomain.lan.
```

Aggiungere tutti i nomi di host presenti nel file forward e salvare le modifiche.

2.5.1 Cosa significa tutto questo

Dal momento che tutto questo è stato aggiunto e che ci si sta preparando a riavviare il nostro server DNS *bind*, esploriamo alcune delle terminologie utilizzate in questi due file.

Far funzionare le cose non è sufficiente se non si conosce il significato di ogni termine, giusto?

- **TTL** sta per "Time To Live". Il TTL indica al server DNS per quanto tempo conservare la cache prima di richiederne una nuova copia. In questo caso, il TTL è l'impostazione predefinita per tutti i record, a meno che non si inserisca manualmente un TTL specifico. L'impostazione predefinita è 86400 secondi o 24 ore.
- **IN** sta per Internet. In questo caso, Internet non viene utilizzato. Consideratela invece come una Intranet.
- **SOA** sta per "Start Of Authority" o per il server DNS primario del dominio
- **NS** sta per "name server"
- **Serial** è il valore utilizzato dal server DNS per verificare che il contenuto del file di zona sia aggiornato
- **Refresh** specifica la frequenza con cui un server DNS slave richiede il trasferimento di una zona dal server master
- **Retry** specifica il tempo di attesa, in secondi, prima di ritentare un trasferimento di zona non riuscito
- **Expire** specifica quanto tempo un server slave aspetterà per rispondere a una query quando il master non è raggiungibile
- **A** È l'indirizzo host o il record di inoltro e si trova solo nel file di inoltro
- **PTR** Il record del puntatore è meglio conosciuto come "reverse " e si trova solo nel nostro file reverse

2.6 Test della configurazione

Una volta creati tutti i file, è necessario assicurarsi che i file di configurazione e le zone siano in ordine prima di riavviare il servizio *bind*.

Controllare la configurazione principale:

```
named-checkconf
```

Questo restituirà un risultato vuoto se tutto è a posto.

Controllare la zona forward:

```
named-checkzone ourdomain.lan /var/named/ourdomain.lan.db
```

Se tutto è a posto, si ottiene un risultato simile a questo:

```
zone ourdomain.lan/IN: loaded serial 2019061800
OK
```

Infine, controllare la zona reverse:

```
named-checkzone 192.168.1.136 /var/named/ourdomain.lan.rev
```

Che restituirà qualcosa di simile se tutto è a posto:

```
zone 192.168.1.136/IN: loaded serial 2019061800
OK
```

Se tutto sembra a posto, riavviare *bind*:

```
systemctl restart named
```

9

2.7 9 usare IPv4 sulla LAN

Per utilizzare SOLO IPv4 sulla LAN, è necessario apportare una modifica in `/etc/sysconfig/named`:

```
vi /etc/sysconfig/named
```

Aggiungete questo alla fine del file:

```
OPTIONS="-4"
```

Salvare le modifiche.

2.8 9 Macchine di prova

È necessario aggiungere il server DNS (nel nostro esempio 192.168.1.136) a ogni macchina che si desidera abbia accesso ai server aggiunti al DNS locale. L'autore mostra solo un esempio di come farlo su una workstation Rocky Linux. Metodi simili esistono per altre distribuzioni Linux, Windows e Mac.

È necessario aggiungere i server DNS all'elenco, non sostituire quelli attualmente presenti, in quanto sarà comunque necessario l'accesso a Internet, che richiederà i server DNS attualmente assegnati. I servizi DHCP (Dynamic Host Configuration Protocol) in genere li assegnano o sono assegnati staticamente.

Si aggiungerà il nostro DNS locale con `nmcli` e poi si riavvierà la connessione.

"Nomi di profilo stupidi"

In NetworkManager, le connessioni non vengono modificate dal nome del dispositivo, ma dal nome del profilo. Può trattarsi di "Connessione cablata 1" o "Connessione wireless 1". È possibile vedere il profilo eseguendo `nmcli` senza alcun parametro:

```
nmcli
```

Verrà visualizzato un risultato come:

```
enp0s3: connected to Wired Connection 1
"Intel 82540EM"
ethernet (e1000), 08:00:27:E4:2D:3D, hw, mtu 1500
ip4 default
inet4 192.168.1.140/24
route4 192.168.1.0/24 metric 100
route4 default via 192.168.1.1 metric 100
inet6 fe80::f511:a91b:90b:d9b9/64
route6 fe80::/64 metric 1024

lo: unmanaged
    "lo"
    loopback (unknown), 00:00:00:00:00:00, sw, mtu 65536

DNS configuration:
    servers: 192.168.1.1
```