RadiusForge

Deployment Guide

Preparing a server, HTTPS, production and upgrades

RadiusForge for ISPs & WISPs · Version 2026.10.17 · radiusforge.co.uk

This guide covers running RadiusForge in production: choosing and preparing a server, networking, HTTPS, the post-install checks worth doing, and safe upgrades. For the basic install itself, see the **Install Guide**.

What you end up with

A single server running **MySQL**, **FreeRADIUS** and the **RadiusForge** web console, reachable at `http://your-server-ip/` (then HTTPS once you set it up). No separate application server is needed — it's the "install it, browse to it" model, done properly with a hardened service.

Choosing the operating system

RadiusForge supports two OS families, and behaves identically on both — only the underlying plumbing differs, which the installer handles.

	Red Hat family	Debian family
Versions	RHEL 8/9, Rocky Linux, AlmaLinux	Ubuntu 22.04 / 24.04 LTS, Debian 12
Installer	`scripts/install-rhel.sh`	`scripts/install-debian.sh`
RADIUS service	`radiusd`	`freeradius`
FreeRADIUS config	`/etc/raddb/`	`/etc/freeradius/3.0/`
Firewall	firewalld	ufw
Subscription needed	RHEL yes; Rocky/Alma no	No

No preference? Ubuntu 24.04 LTS or Rocky Linux are easy starts

Both are free, widely supported, need no subscription, and are long-term-support releases — a sensible default if your organisation has no existing standard.

Preparing the server

- **Give it its own machine.** RadiusForge installs its own FreeRADIUS and MySQL and will not share a server with an existing RADIUS or database server. A dedicated VM is perfect.
- **Size it modestly.** A small server handles a lot of RADIUS traffic. 2 vCPU and 2–4 GB RAM is a comfortable starting point for most ISPs/WISPs; scale up with subscriber count and accounting volume.
- **Networking.** The server needs:
  - **outbound internet** during install (to fetch packages), and
  - to be **reachable** on the address your routers will send RADIUS to, and from the computer you'll manage it from.
- **Fixed address.** Give the server a **static IP** (or a permanent DHCP reservation). Your routers point at this address, so it must not change.

Two network interfaces? Make sure both come back after a reboot

If the server has more than one interface, confirm each is set to connect automatically on boot. The installer enables autoconnect for the interfaces that are up while it runs — verify with the reboot test below.

HTTPS — required before real use

The installer serves plain HTTP so you can log in immediately, but the hotspot **customer portal refuses plain HTTP**, and you should never send staff logins in the clear. Set up a certificate with the bundled script, which covers three cases:

Public server with a real domain — Let's Encrypt (recommended):

```
sudo bash scripts/https-setup.sh letsencrypt \
    --name radius.example.com --email you@example.com --agree-tos
```

This obtains and installs a free, trusted certificate and switches the site to HTTPS. It renews automatically.

A lab or internal test box — self-signed:

```
sudo bash scripts/https-setup.sh self-signed --name radius.lab
```

Browsers will warn (expected for self-signed) — fine for a lab, not for customers.

Your own certificate authority — install a cert you were issued:

```
sudo bash scripts/https-setup.sh install --cert server.crt --chain ca.crt
```

Check what's in use and until when at any time:

```
sudo bash scripts/https-setup.sh status
```

> **Point a DNS name at the server before Let's Encrypt**
> Let's Encrypt validates a real domain name, so create a DNS record (e.g. `radius.example.com`) pointing at the server's public address first. The dashboard's secure-address card warns you well before a certificate is due to expire.

Post-install checks worth doing

Confirm all services are active

```
# Red Hat family:
systemctl is-active radiusforge radiusd nginx mysqld
# Debian family:
systemctl is-active radiusforge freeradius nginx mysql
```

The reboot test (do not skip)

Every service is set to start on boot — but that's only worth anything once you've watched it happen. Reboot and check nothing needed a manual start:

```
sudo reboot
# after a minute, log back in and run:
ip -br addr
systemctl is-active radiusforge nginx    # plus radiusd/mysqld or freeradius/mysql
```

Both interfaces should have their addresses and every service should say `active` with no intervention.

> **Why this matters**
> Two classes of problem only appear on the first reboot: FreeRADIUS starting before the database is ready, and network interfaces not reconnecting. The installer guards against both — the reboot test is how you confirm it on *your* server.

Everyday commands

```
systemctl status radiusforge     # the web application
systemctl restart radiusforge    # after a change
journalctl -u radiusforge -f     # live application logs

# RADIUS service: radiusd (Red Hat) or freeradius (Debian)
systemctl status freeradius
```

FreeRADIUS keeps its own log at `/var/log/radius/radius.log` (Red Hat) or `/var/log/freeradius/radius.log` (Debian).

SELinux note (Red Hat family only)

RHEL, Rocky and Alma ship SELinux in enforcing mode, which adds permission layers beyond normal file permissions. The installer handles the cases RadiusForge needs. If you're troubleshooting an unusual setup and see a "Permission denied" that makes no sense from file permissions alone, run:

```
sudo ausearch -m avc -ts recent
```

It shows exactly what SELinux blocked and why. (Installing under `/opt/radiusforge`, as the installer ensures, avoids the most common case.)

Backups

Your important state is the **MySQL database** (subscribers, plans, sessions, vouchers, staff) and your **configuration**. Take a regular database dump — for example a nightly `mysqldump` of the RadiusForge database to off-server storage — so you can restore after hardware loss. Keep your **licence file** safe too.

> **Upgrades back themselves up**
>
> The upgrade process (below) takes its own backup of the database and configuration before it changes anything — but that's for upgrade safety, not a substitute for your own regular off-server backups.

Upgrading

When a newer release is available, upgrading is one safe command. Download and verify the new release as in the Install Guide, then:

```
sha256sum radiusforge-<new-version>.tar.gz   # compare to the website
tar xzf radiusforge-<new-version>.tar.gz
sudo bash radiusforge-<new-version>/scripts/upgrade.sh
```

The upgrade:

- checks every file against the release manifest before touching anything,
- **backs up** your database and configuration,
- carries your settings across and applies any database changes,
- verifies the result, and **rolls back by itself** if anything isn't right.

Your subscribers keep authenticating throughout.

> **Upgrade during a quiet window the first time**
>
> Although the upgrade is designed to be safe and self-rolling-back, run your first one during a low-traffic window and confirm the reboot test afterwards — standard caution for any production change.

Need a hand planning a production rollout or migration? **info@radiusforge.co.uk**