How to Install TP-Link Omada Software Controller on Ubuntu

TechRounder PDF Edition

Live article: https://www.techrounder.com/how-to/how-to-install-tp-link-omada-software-controller-on-ubuntu/

By Vipin PG | Published July 5, 2024 | Updated April 27, 2026 | Format: Guide | 3 min read

Quick answer

Installing the Omada SDN Controller on Linux requires careful preparation of dependencies-specifically Java, MongoDB, and jsvc-rather than just running the installation package. Successful deployment depends on matching the correct Java version (OpenJDK 17 for newer builds) and ensuring a compatible MongoDB backend is active to avoid service start failures.

Key points

- Critical Dependencies: The controller requires OpenJDK (v17 for latest versions), MongoDB (3.x-7.x), jsvc (for daemon service mode), and curl.
- Installation Process: The recommended workflow involves updating the system, installing base tools and Java, setting up MongoDB, and then installing the controller via a .deb package or tar.gz file.
- Access and Management: Once installed, the controller is accessed via a web browser at ‘https://<server-ip>:8043‘ .
- Common Failure Points: Most installation errors stem from Java version mismatches, inactive MongoDB services, missing jsvc, or dependency conflicts in newer Ubuntu releases.
- Hardware Requirements: The controller is lightweight, typically requiring only a 2-core VM with 2-4 GB of RAM for small to medium deployments.

Most people think installing the Omada controller is just a quick package install. It’s not. The controller itself takes seconds to install, but getting the environment ready is where things usually fall apart.

If you’ve run into a failed service start, MongoDB errors, or Java version conflicts, that’s exactly the issue. The controller is solid. It’s the dependencies that need attention.

What You Actually Need to Install

The Omada controller doesn’t come with its own runtime. You need to set up the system first, or the installer will finish but the service won’t start.

Component: Java (OpenJDK) | Required Version: 8-17+ | Purpose: Runtime for controller | Notes: v5.15+ requires Java 17

Component: MongoDB | Required Version: 3.x - 7.x | Purpose: Database backend | Notes: Not bundled with controller

Component: jsvc | Required Version: Latest | Purpose: Daemon/service runner | Notes: Required for service mode

Component: curl | Required Version: Latest | Purpose: Download & internal scripts | Notes: Light dependency

Data last verified: April 2026

The official Omada Linux install guide confirms that newer controller builds need Java 17, while older versions still work with Java 8. :contentReference[oaicite:2]{index=2}

Step 1 - Get the Ubuntu System Ready

Start with a clean, updated system. Don’t skip this step. Dependency problems often trace back to incomplete upgrades.

```
sudo apt update && sudo apt upgrade -y
```

Install the base tools you’ll need:

```
sudo apt install -y curl jsvc net-tools
```

If you’re setting up a proper homelab stack, this typically goes alongside things like local DNS infrastructure setup for complete network control.

Step 2 - Install Java (Version Matters Here)

This is where a lot of guides get it wrong. Don’t just install Java 8 unless you’re running an older controller version.

For latest Omada Controller (recommended)

```
sudo apt install -y openjdk-17-jdk
```

Check it:

```
java -version
```

According to TP-Link dependency upgrade notes, OpenJDK 17 is now required for newer controller releases. :contentReference[oaicite:3]{index=3}

For older versions (if needed)

```
sudo apt install -y openjdk-8-jre-headless
```

Step 3 - Install MongoDB

The controller relies on MongoDB as its database backend. It won’t start without it.

Install MongoDB (using Ubuntu repo or official repo depending on what your version supports):

```
sudo apt install -y mongodb
```

Older guides still point to MongoDB 3.x, but newer controller builds work with a broader range, including more recent versions. :contentReference[oaicite:4]{index=4}

Community forums mention compatibility issues here pretty often, especially with newer Ubuntu releases. That’s where most installs break, not during the controller installation itself.

Step 4 - Download and Install Omada Controller

Now for the straightforward part.

Grab the latest controller from the official Omada download page and install it.

Using .deb package (recommended for Ubuntu)

```
sudo dpkg -i Omada_SDN_Controller_vX.X.X_linux_x64.deb
```

This is the cleanest method and works well with system services. :contentReference[oaicite:5]{index=5}

Alternative: tar.gz method

```
tar zxvf Omada_Controller_vX.X.X_linux_x64.tar.gz
cd Omada_Controller_*
sudo bash install.sh
```

The official installation steps confirm both methods work across Ubuntu and Debian systems. :contentReference[oaicite:6]{index=6}

Step 5 - Start and Access the Controller

After installation, the service should start on its own. If it doesn't:

```
sudo systemctl start omada
sudo systemctl enable omada
```

Access the web interface at:

```
https://<server-ip>:8043
```

Initial setup happens through a browser wizard where you adopt your devices.

If you're building out a full self-hosted stack, this typically runs alongside tools like home server container setup for better isolation and easier scaling.

Common Issues That Break Installs

1. Java mismatch

Controller installs without errors but the service fails quietly. Check your Java version first.

2. MongoDB not running

If MongoDB isn't active, the controller can't initialize its database.

3. Missing jsvc

Without jsvc, the service won't run as a daemon properly. TP-Link lists it as required. :contentReference[oaicite:7]{index=7}

4. Ubuntu version conflicts

Newer Ubuntu releases sometimes break older MongoDB dependencies. This is where most forum complaints originate.

Some users get around this by pinning MongoDB versions or switching to containerized installs.

Running It in a Homelab Environment

In a real setup, the controller rarely runs by itself. It usually sits in a stack with DNS, monitoring, and routing layers.

For instance, pairing it with secure DNS proxy setup gives you better control over network traffic and filtering.

Worth mentioning: the controller doesn't need heavy resources. A 2-core VM with 2-4 GB RAM handles small to medium deployments without issues.

What to Do Next

Once the controller is running, adopt your devices and test roaming, VLANs, and guest networks. That’s where Omada really shows what it can do. Keep an eye on dependency updates, especially Java and MongoDB, because future controller upgrades often depend on them.

References

1. tp-link.com - nl-be / support - https://www.tp-link.com/nl-be/support/faq/3272/
2. tp-link.com - us / support - https://www.tp-link.com/us/support/faq/4397/
3. tp-link.com - us / support - https://www.tp-link.com/us/support/download/omada-software-controller/
4. support.omadanetworks.com - us / document - https://support.omadanetworks.com/us/document/13088/