LinuxServers

This is the book that house documentation on all the Linux Servers

    - Installing Wazuh Central Components Natively on Ubuntu Server
    - Managing Users and Roles in Wazuh (v4.14+)
    - Installing Wazuh Agent on Linux (Ubuntu/Debian)

- Peanut
    - How to Connect PeaNUT (CasaOS) to a Synology UPS

- Matomo
    - Setting Up Matomo Analytics with a Dedicated MariaDB Instance
    - Integrating Matomo Analytics with BookStack
    - Integrating Nextcloud Hub 28 with Matomo Analytics

- HestiaCP
- NordVPN
    - NordVPN Installation and Strict Configuration on Lubuntu

Nextcloud

Nextcloud

High-Performance Nextcloud & MariaDB Deployment (CasaOS)

Status: Verified Deployment
Environment: Ubuntu (CasaOS) | Synology DS1522+ Backend
Core Architecture: Dedicated Mini PC for compute; "GoonersNAS" for muscle storage.

---

1. Preparation: The Network Bridge

To allow Nextcloud and MariaDB to communicate securely without using public IP addresses, they must reside on the same Docker network.

- **Network Name:** `big-bear-nextcloud-ls`
- **Note:** Ensure both containers are assigned to this network in their respective CasaOS settings to enable internal DNS resolution.

---

2. Database Setup (MariaDB)

We use the LinuxServer image for consistency and PUID/PGID support, which is vital for permission management.

Category	Setting / Value
Image	`linuxserver/mariadb`
Network	`big-bear-nextcloud-ls`
Port	Host 3306 -> Container 3306
Volume	`/DATA/AppData/mariadb/config` -> `/config`

Environment Variables:

```
MYSQL_ROOT_PASSWORD = [YourSecretRootPassword]
MYSQL_DATABASE = nextcloud
MYSQL_USER = nc_user
MYSQL_PASSWORD = [YourSecretUserPassword]
PUID = 1000
PGID = 1000
```

```
TZ = America/Chicago
```

3. Application Setup (Nextcloud LS)

The **Nextcloud LS** (LinuxServer.io) version is preferred for its robust handling of PUID/PGID variables when mapping to NAS-based storage.

Category	Setting / Value
Image	`linuxserver/nextcloud`
Port	Host 10443 -> Container 443
Web UI	HTTPS (Required for Authentik)
Volume 1	`/DATA/AppData/nextcloud/config` -> `/config`
Volume 2 (NAS)	`/mnt/synology/nextcloud_data` -> `/data`

4. Initial Configuration (The Setup Screen)

Navigate to `https://nc.goonersnas.com` to complete the wizard.

- **Admin Account:** Create a master local admin.
- **Storage & Database:** Select **MySQL/MariaDB**.
- **Database User:** `nc_user`
- **Database Host:** `mariadb:3306`

5. Troubleshooting & Critical Fixes

> **The "Access Denied" Loop:** If the installer rejects your credentials, MariaDB likely initialized with old variables.
> **Fix:** Stop MariaDB, delete all contents of `/DATA/AppData/mariadb/config`, and restart. It will re-generate the database with the correct PUID/PGID and credentials.

> **The "App Unavailable" Error:** This is a protocol mismatch in CasaOS.
> **Fix:** Manually use `https://` in the browser and update the App Settings to use

> the HTTPS dropdown.

---

6. Post-Install Checklist

- [ ] Enable **External Storage Support** for Synology integration.
- [ ] Install **Social Login** for Authentik OIDC.
- [ ] Download **ONLYOFFICE Desktop Editors** for native local editing.

Nextcloud

Installing ONLYOFFICE on Nextcloud (CasaOS/Ubuntu)

Overview

This guide covers deploying **ONLYOFFICE Document Server** as a backend engine via CasaOS and integrating it with **Nextcloud**. This setup provides a high-performance office suite capable of handling Microsoft Office formats (`.docx`, `.xlsx`, `.pptx`) directly within the browser.

Prerequisites

- **Nextcloud** running on Ubuntu (via Docker/CasaOS).
- **Reverse Proxy** already configured for your main domain (`nc.goonersnas.com`).
- A dedicated subdomain (e.g., `office.goonersnas.com`) with an SSL certificate.

Step 1: Deploy ONLYOFFICE Document Server (Backend)

We use a separate Docker container for the rendering engine to ensure stability and performance.

1. Open **CasaOS** and click the **+** icon to **install a customized app**.
2. Click **Import** in the top right and paste the following Docker Compose snippet:

```
name: onlyoffice
services:
  onlyoffice:
    image: onlyoffice/documentserver:latest
    container_name: onlyoffice-docs
    environment:
      - JWT_ENABLED=true
      - JWT_SECRET=SecretGoonersKey123
    ports:
      - "8888:80"
    restart: unless-stopped
```

3. **Submit** and **Install**.

4. *Optional:* Change the **Network** setting in CasaOS to match your Nextcloud bridge (e.g., `big-bear-nextcloud-ls`).

Step 2: Reverse Proxy Configuration (Crucial)

To avoid the **Mixed Active Content** error, ONLYOFFICE must be served over HTTPS.

- Point your subdomain (e.g., `office.goonersnas.com`) to your server IP.
- In your Reverse Proxy manager, create a new host:
  - **Domain:** `office.goonersnas.com`
  - **Forward IP:** `192.168.0.152`
  - **Forward Port:** `8888`
- Enable **SSL** and force SSL.

Step 3: Integration in Nextcloud

1. **Install the App:** Go to **Apps** > **Office & text**. Find **ONLYOFFICE, click Download, and enable**.
2. **Configure Settings:**
   - Go to **Administration settings** > **ONLYOFFICE**.
   - **Document Editing Service address:** `https://office.goonersnas.com/`
   - **Secret key:** `SecretGoonersKey123`
   - Click **Save**.

> **IMPORTANT:** If you see a red error banner stating "Mixed Active Content is not allowed," it means your Nextcloud (`https`) is trying to talk to an insecure ONLYOFFICE address (`http`). **Always** use the secure public subdomain URL in the Nextcloud settings to resolve this.

Verification

To verify the installation, navigate to your **Files** app and create a new **Document**. If the editor loads the Word-style interface, the integration is successful.

Beszel

Beszel Agent Installation (Linux)

This Knowledge Base (KB) article is designed for internal documentation[cite: 1]. It outlines the streamlined "nuke and pave" installation method, perfected for Ubuntu Server environments, ensuring a clean deployment every time

> - **Target Systems:** Ubuntu Server / Debian-based distributions
> - **Hardware Optimization:** 7th-Gen Intel Core i5 | 16GB DDR4 | NVMe SSD
> - **Default Port:** 45876

---

1. Prerequisites

Before beginning the installation, verify the following requirements:

- **Connectivity:** Ensure the target system can communicate with the Beszel Hub (CasaOS) on the local network[cite: 8].
- **Static IP:** Confirm the target server has a DHCP reservation or a static IP assigned[cite: 9].
- **Firewall:** Port 45876 must be open for inbound TCP traffic[cite: 10].

2. Installation Command (Master)

The following command uses the official Beszel automated script [cite: 11, 12]. This process creates a dedicated unprivileged `beszel` user downloads the latest binary and configures the systemd service with your specific public key[cite: 12].

```
curl -sL https://get.beszel.dev -o /tmp/install-agent.sh && \
chmod +x /tmp/install-agent.sh && \
sudo /tmp/install-agent.sh -p 45876 -k "ssh-ed25519
AAAAC3NzaC1lZDI1NTE5AAAAICJ7lFlWxcv1b25gymPNRAvp0ptAJChTuNYvmnomZpFW"
```

3. Post-Installation Configuration

Immediately after running the installation script, execute these commands to ensure network stability and service health

A. Open Firewall (UFW)

```
sudo ufw allow 45876/tcp
sudo ufw reload
```

B. Verify Service Status

Ensure the agent is active and not stuck in a crash loop

```
sudo systemctl status beszel-agent.service
```

4. Troubleshooting & Maintenance

Clean Uninstall (Nuke & Pave)

If the agent fails to start due to a malformed key or configuration error, run this sequence to completely purge the installation before retrying Step 2

```
sudo systemctl stop beszel-agent.service
sudo rm -f /etc/systemd/system/beszel-agent.service
sudo rm -rf /opt/beszel-agent
sudo userdel beszel
sudo systemctl daemon-reload
```

Source: [cite: 28]

Log Inspection

To view live logs for connection rejections or hardware sensor errors

```
sudo journalctl -u beszel-agent.service -f --no-pager
```

NVMe Health Note

The Beszel agent has a negligible I/O footprint[cite: 34]. To maintain the performance of your 256GB NVMe SSD, ensure the `fstrim` timer is active alongside the agent

```
sudo systemctl enable --now fstrim.timer
```

Purging and Reinstalling Beszel Agent (Linux)

This Knowledge Base (KB) article documents the **"Nuke and Pave"** method. This is the most reliable way to resolve "Invalid SSH Key" errors, service crashes, or corrupted configurations on your Ubuntu Server instances.

Scenario: Use this procedure if the Beszel Hub shows a red "Offline" dot despite the service running, or if `journalctl` reports *failed to parse key* or *illegal base64 data*.

Logic: This method forcefully terminates lingering processes, clears the systemd memory cache, and deletes the unprivileged user to ensure the new installation starts from a completely "zeroed" state.

Phase 1: The "Nuke" (Complete Removal)

Run this block to obliterate the existing installation. This is safe to run even if some files are already missing.

```
# 1. Force kill any hung agent processes
sudo killall -9 beszel-agent 2>/dev/null

# 2. Stop and disable the service unit
sudo systemctl stop beszel-agent.service 2>/dev/null
sudo systemctl disable beszel-agent.service 2>/dev/null

# 3. Delete the service configuration and binary files
sudo rm -f /etc/systemd/system/beszel-agent.service
sudo rm -rf /opt/beszel-agent

# 4. Remove the dedicated agent user
sudo userdel beszel 2>/dev/null

# 5. Flush the systemd daemon cache
sudo systemctl daemon-reload
```

Phase 2: The "Pave" (Clean Reinstall)

Execute the master installation command. This uses your verified Public Key to ensure an immediate handshake with the CasaOS Hub.

```
curl -sL https://get.beszel.dev -o /tmp/install-agent.sh && \
chmod +x /tmp/install-agent.sh && \
sudo /tmp/install-agent.sh -p 45876 -k "ssh-ed25519
AAAAC3NzaC1lZDI1NTE5AAAAICJ7lFlWxcv1b25gymPNRAvp0ptAJChTuNYvmnomZpFW"
```

Phase 3: Post-Flight Verification

After the script reports success, verify that the service is active and that the connection is established.

1. Check Service Heartbeat

```
sudo systemctl status beszel-agent.service
```

Target: `Active: active (running)`

2. Verify Hub Communication

```
sudo journalctl -u beszel-agent.service -n 20 --no-pager
```

- **Target:** You should see `INFO Starting SSH server addr=:45876`.
- **Target:** You should **not** see any `WARN` messages regarding "Invalid SSH key."

Admin Tips for the Micro PC Environment

- **Hostname Resolution:** If you see *sudo: unable to resolve host*, ensure your `/etc/hosts` file matches your hostname:
  `echo "127.0.1.1 $(hostname)" | sudo tee -a /etc/hosts`
- **NVMe Lifespan:** By using the "Nuke and Pave" method only when troubleshooting, you avoid unnecessary write cycles on your 512GB NVMe SSD. The agent itself runs primarily in RAM (approx. 6-10MB).
- **Remote Management:** If a server becomes unreachable during this process, use your **JetKVM** to access the tty console directly and verify the network interface with `ip a`.

Next Step: Your Linux recovery documentation is complete. Shall we proceed with the Windows 11 agent installation using the NSSM service manager method?

Beszel & Gotify Integration

This guide outlines the process for integrating **Beszel** with a self-hosted **Gotify** instance to receive real-time server health alerts. This setup ensures your notification data remains within your own infrastructure while providing reliable push notifications to mobile devices.

Overview

Beszel utilizes the **Shoutrrr** library to handle notifications, allowing it to send alerts to various services via a standardized URL format. By connecting it to Gotify, you gain a private, lightweight notification server that operates independently of third-party cloud providers.

Step 1: Create a Gotify Application

To allow Beszel to communicate with Gotify, you must generate a unique API token.

1. Log in to your **Gotify Web UI** (e.g., `https://gotify.yourdomain.com`).
2. Navigate to the **Apps** tab in the top header.
3. Click **Create Application**.
4. Name the application (e.g., "Beszel Alerts") and click **Create**.
5. **Copy the Token:** A string of characters will appear in the "Token" column. Save this for the next step.

Step 2: Configure Beszel Notification URL

Now, provide Beszel with the connection string for your Gotify server.

1. Open your **Beszel Dashboard**.
2. Go to **Settings** > **Notifications**.
3. Under the **Webhook / Push notifications** section, click **+ Add URL**.
4. Enter the URL using the following format (replace the placeholders with your actual domain and token):

```
gotify://godify.yourdomain.com/YOUR_APP_TOKEN_HERE
```

Note: If testing strictly on a local network without a reverse proxy, use `gotify://[LOCAL_IP]:[PORT]/[TOKEN]`.

5. Click **Save Settings**.
6. Click **Test URL** to confirm a "Test Message" appears in your Gotify dashboard.

Step 3: Enable Alert Thresholds

Notifications are only dispatched when specific triggers are met.

1. On the Beszel **All Systems** page, click the **Bell Icon (□□)** next to the server you wish to monitor.
2. Toggle **Status** to "ON" (alerts you if the server goes offline).
3. Set thresholds for **CPU**, **Memory**, or **Disk Usage** (e.g., "Average exceeds 90% for 10 minutes").
4. Click **Save**.

Step 4: Mobile Integration

To receive these alerts as push notifications on your mobile device, link the mobile app to your server.

For Android:

- Download **Gotify** from the Play Store or F-Droid.
- Enter your server URL: `https://godify.yourdomain.com`.
- Log in with your Gotify user credentials.
- **Crucial:** Navigate to phone **Settings > Apps > Gotify > Battery** and set it to **Unrestricted** to prevent the OS from killing the background process.

For iOS:

- Download **iGotify** from the App Store.
- Enter your server URL and login credentials.
- Follow the in-app prompts to enable push certificates.

Security Best Practices

- **SSL/HTTPS:** Always route Gotify through a reverse proxy (like Nginx Proxy Manager) with a valid SSL certificate.
- **WebSockets:** Ensure your reverse proxy has "Websockets Support" enabled, as Gotify relies on them for real-time delivery.
- **Privacy:** Avoid sharing your App Tokens publicly, as they allow any service to push messages to your devices.

PiHole

PiHole

High-Availability Pi-hole Synchronization & Docker Networking

Last Updated: March 2026 | **Target Hardware:** Micro PC Cluster (Intel i5 7th-gen)

Overview: This article documents the validated procedure for deploying a high-availability DNS sinkhole architecture using two micro PCs running CasaOS. [cite: 1, 3, 7] By leveraging **Gravity Sync**, the Secondary Pi-hole (192.168.x.x) mirrors the DNS payload of the Primary Pi-hole (192.168.x.x).

Prerequisites

- **Primary Server (LinuxServer):** 192.168.x.x
- **Secondary Server (LinuxServer2):** 192.168.x.x
- Pi-hole installed via Docker (CasaOS) on both nodes.
- Active SSH credentials for both servers.

Step 1: Install Gravity Sync on Both Servers

The synchronization tool must be installed on the host OS of **both** micro PCs. [cite: 16] To bypass retired domain links, execute the following command on both nodes:

```
curl -sSL https://raw.githubusercontent.com/vmstan/gs-install/main/gs-install.sh | bash
```

Step 2: Configure the Secondary Node

Because the Secondary Server "pulls" data from the Primary, execute the configuration wizard from **LinuxServer2 (192.168.0.152)**: [cite: 21]

1. **Initiate Wizard:** Run `gravity-sync config`.
2. **Define Remote Host:** Enter the IP of the Primary Server (192.168.x.x
3. **Authenticate:** Enter the Primary Server's SSH username and password to register RSA keys.

4. **Define Container Names:** Manually confirm the container names `pihole` for both Local and Remote prompts.

Step 3: Perform Initial Manual Sync

On the **Secondary Server (192.168.x.x)**, run the following to establish the baseline mirror:

```
gravity-sync pull
```

Step 4: Automate the Synchronization

To ensure future changes to blocklists or local DNS are synced automatically, enable the background service on the Secondary Server:

```
gravity-sync auto
```

Note: This registers a systemd timer that compares database hashes every 5 minutes with zero noticeable overhead on NVMe drives. [cite: 38, 39]

Step 5: Resolve Docker Network Restrictions (Pi-hole v6 Fix)

By default, CasaOS Docker deployments may cause Pi-hole to drop incoming LAN requests. [cite: 42] You must manually configure the secondary node to trust LAN traffic:.

- Log into the Secondary Web UI: `http://192.168.x.x:81/admin`
- Navigate to **Settings > DNS.**
- Toggle the top-right corner from **Basic** to **Advanced**.
- In **Interface settings**, select **Permit all origins**.
- Click **Save & Apply**.

Step 6: Validate Failover

Perform a direct query test from a Windows client on the same network: [cite: 51]

```
nslookup google.com 192.168.x.x
```

Expected Result: The terminal should return a list of IP addresses. [cite: 54] If it returns "Timeout," re-verify the settings in Step 5.

Bookstack

Deploying BookStack via CasaOS App Store

Environment Context:

- **Hostname:** LinuxServer
- **OS:** Ubuntu Server (CasaOS UI layer)
- **Hardware:** Micro PC (7th-Gen Intel Core i5, 16GB DDR4 RAM, 512GB NVMe SSD)
- **Service:** BookStack (BigBearCasaOS Repository)

---

1. Host OS Optimization

Before deploying database-heavy containers, ensure the host Ubuntu Server is configured to maintain the write endurance and performance of the 512GB NVMe SSD.

Connect to `LinuxServer` via SSH (or via the local KVM console) and enable the continuous TRIM timer:

```
sudo systemctl enable --now fstrim.timer
```

2. CasaOS Installation

To prevent volume overlap and network isolation issues, utilize the pre-configured application template from the CasaOS App Store.

1. Navigate to the CasaOS web dashboard (`http://192.168.x.x`).
2. Open the **App Store**.
3. Search for and select **BookStack** (Developer: Dan Brown / BigBearCasaOS).
4. Click **Install**.
5. **Crucial:** Once the installation finishes, *do not click the app icon yet*. Proceed immediately to configuration.

3. Configuration & Routing Fixes

The default BigBear template contains a placeholder URL and uses port `8080`, which often causes conflicts and results in a `500 Internal Server Error`. These must be adjusted before the first boot.

1. Click the three vertical dots (⋮) on the newly installed BookStack icon and select **Settings**.
2. **Update Web UI Port:** At the top of the settings window, change the Web UI port `8080` to a dedicated port (e.g., `6875`).
3. **Update Host Port:** Scroll down to the **Port** section. Change the Host port `8080` to `6875`. *(Leave the Container port as 80)*.
4. **Fix App URL:** Scroll down to the **Environment Variables** section. Locate the `APP_URL` key.
5. Change the value from the literal placeholder `http://[change to your IP]:8080` to the server's exact static IP and new port:
   - **Value:** `http://192.168.x.x:6875`
6. Click the blue **Save** button.

4. Database Initialization

When the settings are saved, CasaOS will restart the container. BookStack's internal MariaDB container must now generate its base tables on the NVMe drive.

- **Wait exactly 60 to 120 seconds** to allow the i5 processor to complete the initial database migrations.
- Attempting to access the application before this completes may result in an incomplete database generation and a fatal crash loop.

5. First Login and Security Hook

Once initialization is complete, click the BookStack app icon on the CasaOS dashboard. You will be routed to the login screen.

- **Default Email:** `admin@admin.com`
- **Default Password:** `password`

> **Immediate Post-Install Action:** Upon successful login, navigate to the administrator profile in the top right corner. Select **Edit Profile**, update the email address to your personal address, and change the password to a secure string to lock down the instance.

LibreSpeed

Base: Installing & Troubleshooting LibreSpeed on CasaOS

Objective: Set up a self-hosted LibreSpeed server on CasaOS to perform local network speed tests, bypassing internet variables to test actual LAN/Wi-Fi throughput.

Environment:

- **Host OS:** Ubuntu Server
- **Management UI:** CasaOS
- **Application:** LibreSpeed (`ghcr.io/librespeed/speedtest`)

Part 1: Installation via CasaOS

Since port 80 is often occupied by other services (like Nginx Proxy Manager), this guide uses port **8082** for the host interface.

Method A: YAML Import (Recommended & Fastest)

1. Open your **CasaOS Dashboard**.
2. Click the **App Store** icon, then click **Custom Install** in the top right corner.
3. Click the **Import** icon (document with an arrow) in the upper right.
4. Paste the following configuration:

```
services:
  librespeed:
    image: ghcr.io/librespeed/speedtest
    container_name: librespeed
    ports:
      - "8082:8080"
```

```
    environment:
      - MODE=standalone
      - TELEMETRY=false
    restart: always
```

5. Click **Next**, verify the auto-filled settings, and click **Install**.

Method B: Manual Configuration

If manually entering the details in the **Custom Install** menu, use the following parameters:

- **Docker Image:** `ghcr.io/librespeed/speedtest`
- **Title:** `LibreSpeed`
- **Web UI:** `http://[Server_IP]:8082/`
- **Port:** Host `8082` | Container `8080` | Protocol `TCP`
- **Environment Variables:**
  - Key: `MODE` | Value: `standalone`
  - Key: `TELEMETRY` | Value: `false`

Part 2: Troubleshooting "Connection Refused"

If you attempt to access `http://<Server_IP>:8082` and receive a **"This site can’t be reached / Connection Refused"** error, follow these diagnostic steps.

Step 1: Verify Host Firewall (UFW) Status

Ubuntu's Uncomplicated Firewall (UFW) may block newly assigned ports. Check and open the port using the terminal:

```
sudo ufw allow 8082/tcp
sudo ufw reload
```

(Note: If the terminal returns `Firewall not enabled (skipping reload)`, UFW is turned off and is not the cause of the blockage).

Step 2: Test Local Container Connectivity

Determine if the container is actively listening on the host by running a local HTTP header request from the server terminal:

```
curl -I http://localhost:8082
```

- **Result `HTTP/1.1 200 OK`:** The container is functioning properly. The issue is likely network-wide firewall or router isolation.
- **Result `Connection reset by peer`:** The container is receiving the request but the internal application is crashing or rejecting the traffic. Proceed to Step 3.

Step 3: Inspect Container Logs for Internal Port Binding

If the connection is being reset by the peer, the internal application inside the Docker container is likely misconfigured.

1. In CasaOS, click the three dots (⋮) on the LibreSpeed app and select **Settings**.
2. Click the **Terminal/Logs** icon (`>_`) in the top right and view the **Logs** tab.
3. Look for the Apache port configuration sequence. You may see lines like this:

```
+ '[' 8080 '!=' 80 ']'
+ sed -i 's/^Listen 80$/Listen 8080/g' /etc/apache2/ports.conf
+ sed -i 's/*:80>/*:8080>/g' /etc/apache2/sites-available/000-default.conf
```

The Root Cause: Recent versions of the `ghcr.io/librespeed/speedtest` image have updated their internal Apache web server to listen on port `8080` by default, rather than the standard port `80`. If CasaOS is configured to route traffic to container port `80`, the traffic hits a dead end, resulting in the "Connection reset by peer" error.

The Fix:

1. Open the LibreSpeed **Settings** in CasaOS.
2. Scroll to the **Port** section.
3. Change the **Container** port from `80` to **`8080`**.
4. Click **Save** to restart the container with the correct mapping.

LibreSpeed

Installing & Troubleshooting LibreSpeed on CasaOS

Objective: Set up a self-hosted LibreSpeed server on CasaOS to perform local network speed tests, bypassing internet variables to test actual LAN/Wi-Fi throughput.

Environment:

- **Host OS:** Ubuntu Server
- **Management UI:** CasaOS
- **Application:** LibreSpeed (`ghcr.io/librespeed/speedtest`)

Part 1: Installation via CasaOS

Since port 80 is often occupied by other services (like Nginx Proxy Manager), this guide uses port **8082** for the host interface.

Method A: YAML Import (Recommended & Fastest)

1. Open your **CasaOS Dashboard**.
2. Click the **App Store** icon, then click **Custom Install** in the top right corner.
3. Click the **Import** icon (document with an arrow) in the upper right.
4. Paste the following configuration:

```
services:
  librespeed:
    image: ghcr.io/librespeed/speedtest
    container_name: librespeed
    ports:
      - "8082:8080"
    environment:
      - MODE=standalone
      - TELEMETRY=false
```

```
    restart: always
```

5. Click **Next**, verify the auto-filled settings, and click **Install**.

Method B: Manual Configuration

If manually entering the details in the **Custom Install** menu, use the following parameters:

- **Docker Image:** `ghcr.io/librespeed/speedtest`
- **Title:** `LibreSpeed`
- **Web UI:** `http://[Server_IP]:8082/`
- **Port:** Host `8082` | Container `8080` | Protocol `TCP`
- **Environment Variables:**
  - Key: `MODE` | Value: `standalone`
  - Key: `TELEMETRY` | Value: `false`

Part 2: Troubleshooting "Connection Refused"

If you attempt to access `http://<Server_IP>:8082` and receive a **"This site can’t be reached / Connection Refused"** error, follow these diagnostic steps.

Step 1: Verify Host Firewall (UFW) Status

Ubuntu's Uncomplicated Firewall (UFW) may block newly assigned ports. Check and open the port using the terminal:

```
sudo ufw allow 8082/tcp
sudo ufw reload
```

(Note: If the terminal returns `Firewall not enabled (skipping reload)UFW is turned off and is not the cause of the blockage.`

Step 2: Test Local Container Connectivity

Determine if the container is actively listening on the host by running a local HTTP header request from the server terminal:

```
curl -I http://localhost:8082
```

- **Result** `HTTP/1.1 200 OK`**:** The container is functioning properly. The issue is likely due to a network-wide firewall or router isolation.
- **Result** `Connection reset by peer`The container is receiving the request, but the internal application is crashing or rejecting the traffic. Proceed to Step 3.

Step 3: Inspect Container Logs for Internal Port Binding

If the connection is being reset by the peer, the internal application inside the Docker container is likely misconfigured.

1. In CasaOS, click the three dots (⋮) on the LibreSpeed app and select **Settings**.
2. Click the **Terminal/Logs** icon (`>_`) in the top right and view the **Logs** tab.
3. Look for the Apache port configuration sequence. You may see lines like this:

```
+ '[' 8080 '!=' 80 ']'
+ sed -i 's/^Listen 80$/Listen 8080/g' /etc/apache2/ports.conf
+ sed -i 's/*:80>/*:8080>/g' /etc/apache2/sites-available/000-default.conf
```

The Root Cause: Recent versions of the `ghcr.io/librespeed/speedtest` Image has updated its internal Apache web server to listen on port `8080` by default, rather than the standard port `80`. If CasaOS is configured to route traffic to the container port `80The traffic hits a dead end, resulting in the "Connection reset by peer" error.`

The Fix:

1. Open the LibreSpeed **Settings** in CasaOS.
2. Scroll to the **Port** section.
3. Change the **Container** port from `80` to `8080`.
4. Click **Save** to restart the container with the correct mapping.

Cockpit

Cockpit

Setting Up Cockpit for Multi-Server Management (Linux Mint)

This knowledge base article summarizes the steps taken to get Cockpit running, including specific fixes for errors encountered.

Objective

To install Cockpit on a primary Linux Mint server and connect other Linux Mint servers to manage them from a single web interface.

Prerequisites

- **Primary Server:** Linux Mint XFCE (e.g., LinuxServer)
- **Secondary Servers:** Linux Mint XFCE (e.g., MediaServer, LinuxServer3)
- **Unsupported:** Home Assistant OS (HAOS) cannot host Cockpit directly.

Step 1: Install Cockpit on the Primary Server

Run these commands on the computer you want to use as your main controller:

1. **Install the software:**

```
sudo apt update && sudo apt install cockpit -y
```

2. **Enable the service:**

```
sudo systemctl enable --now cockpit.socket
```

3. **Open the firewall:**

```
sudo ufw allow 9090
```

Step 2: Troubleshooting Installation Errors

Issue: `apt` failed due to a broken "synosnap" (Synology Active Backup) package.

Fix: Use a dummy script to trick the system into removing the broken package.

1. **Create the dummy script structure:**

```
sudo mkdir -p /opt/synosnap/hooks/
sudo touch /opt/synosnap/hooks/dpkg-post-hook.sh
sudo chmod +x /opt/synosnap/hooks/dpkg-post-hook.sh
```

2. **Force the install to finish:**

```
sudo apt -f install
```

3. **Purge the broken package:**

```
sudo dpkg --purge synosnap
```

Step 3: Prepare Secondary Servers

Issue: "Unable to contact... port 22" when adding a host.
Cause: Linux Mint does not have the SSH server enabled by default.

Run these commands on **every** secondary server you want to control:

1. **Install and enable SSH:**

```
sudo apt install openssh-server -y
sudo systemctl enable --now ssh
```

2. **Allow SSH through firewall:**

```
sudo ufw allow ssh
```

3. **Install Cockpit packages (for metrics):**

```
sudo apt install cockpit cockpit-pcp -y
```

Step 4: Connect Servers in the GUI

1. **Access the Web Interface:** Open a browser and go to `https://<PRIMARY_IP>:9090`.
2. **Note:** Accept the "Not Secure" / SSL warning (Advanced -> Proceed).
3. **Log in:** Use your Linux username and password.
4. **Add Host:**
   - Click the dropdown menu in the top-left (or "Overview").
   - Click "Add new host".
   - Enter the IP of the secondary server.
   - Check "Log in automatically" to save credentials.

Step 5: Enabling Metrics & History

Issue: Missing graphs/history.
Fix: The `cockpit-dashboard` package is deprecated. Use PCP (Performance Co-Pilot) instead.

Run this on **all** servers to enable history recording:

```
sudo apt install cockpit-pcp -y
sudo systemctl enable --now pmlogger
sudo systemctl enable --now pmproxy
```

Known Limitations

- **Home Assistant:** Cannot be added to Cockpit (unless running as a VM inside Linux).
- **"Single Pane" Dashboard:** The old grid view of all servers at once has been deprecated.
- **Navigation:** Use the dropdown menu to switch between servers.

Records

Authentik

Installing Authentik on CasaOS (Big Bear Template)

Overview

This article outlines the successful installation of **Authentik** on an Ubuntu server running **CasaOS** (192.168.0.152). Authentik is a complex application consisting of four separate containers that must communicate over a shared internal network.

---

1. Pre-Installation Cleanup

To prevent database corruption or "ghost" network conflicts, start with a clean environment.

- **Remove Existing Apps:** Uninstall any previous Authentik or Postgres attempts via the CasaOS dashboard.
- **Clear AppData:** Use the Files app to delete the `AppData/big-bear-authentik` folder.
- **Prune Ghost Networks:** If an installation fails with a "Label" or "Network" error, use a terminal to run:

```
docker network prune
```

(Enter 'y' when prompted to remove unused custom networks.)

---

2. Installation via Big Bear App Store

Navigate to the App Store in CasaOS and select **Authentik** from the Big Bear repository. Before clicking "Install," you must adjust the settings for all four components.

A. Component: big-bear-authentik (Server)

- **Network:** Select `big-bear-authentik` (or the longest available big-bear network name).
- **Ports:** Add a new port mapping:

    - **Host:** 9000
    - **Container:** 9000
    - **Protocol:** TCP
- **Web UI:** Ensure the URL is set to `http://[IP]:9000`.

B. Components: DB, Redis, and Worker

Navigate to the tabs for `big-bear-authentik-db`, `big-bear-authentik-redis`, and `big-bear-authentik-worker`.

- **Crucial Step:** Change the **Network** on every single tab to match the one selected for the Server (`big-bear-authentik`).
- **Internal Communication:** Do **not** map host ports for the DB or Redis; they communicate internally over the shared bridge network.

---

3. Post-Installation & Troubleshooting

The "Authentik Starting" Screen

Upon first boot, the server will display a black screen stating "authentik starting."

- **Cause:** The server is running initial database migrations and building internal tables.
- **Action:** Wait **3–5 minutes**. Do not restart the containers during this phase.

Database "Unhealthy" Status

If the DB is marked unhealthy, it is usually due to a password mismatch in the environment variables.

- **Fix:** Ensure `POSTGRES_PASSWORD` in the DB tab matches the `AUTHENTIK_POSTGRESQL__PASSWORD` in both the Server and Worker tabs.

---

4. Initial Admin Account Setup

Authentik does not ship with a default password for the `akadmin` user. You must trigger the initial setup flow.

1. Navigate to: `http://192.168.0.152:9000/if/flow/initial-setup/`
2. Set a strong master password for the **akadmin** account.

3. Log in at the standard portal using:
    - **Username:** akadmin
    - **Password:** (The one you just created)

Summary of "What Worked"

Requirement	Value / Selection
Shared Network	big-bear-authentik (Set on all 4 tabs)
Primary Port	9000 (Mapped for HTTP access)
Worker Command	worker
Setup URL	/if/flow/initial-setup/

Linking Authentik SSO to Nextcloud (OIDC)

> **Purpose:** This guide outlines the steps to enable Single Sign-On (SSO) for Nextcloud using Authentik via the OpenID Connect (OIDC) protocol. This allows users to log in to Nextcloud using their central Authentik credentials.

Prerequisites

- A working **Authentik** instance (e.g., `auth.goonersnas.com`).
- A working **Nextcloud** instance (e.g., `nc.goonersnas.com`).
- Administrator access to both platforms.

Step 1: Create the Authentik Provider

The Provider acts as the authentication engine for the handshake.

1. Navigate to **Applications > Providers** in the Authentik Admin interface.

2. Click **Create** and select **OAuth2/OpenID Provider**.
3. Set the following values:

Field	Value
Name	Nextcloud
Authentication flow	default-authentication-flow
Authorization flow	default-provider-authorization-implicit-consent
Client Type	Confidential

4. In the **Redirect URIs** section, add the following (adjust domain as needed):

```
https://nc.goonersnas.com/index.php/apps/sociallogin/custom_oidc/authentik
```

Click **Finish** and then copy your **Client ID** and **Client Secret** from the provider details page.

Step 2: Create the Authentik Application

The Application links the Provider to a user-facing icon.

1. Navigate to **Applications > Applications**.
2. Click **Create**.
3. **Name:** Nextcloud | **Slug:** nextcloud
4. **Provider:** Select the "Nextcloud" provider created in Step 1.
5. Click **Finish**.

Step 3: Configure Nextcloud Social Login

Install the **Social Login** app from the Nextcloud App Store, then navigate to **Settings > Administration > Social login**.

3.1 General Settings

Check the following boxes at the top of the page:

- **Uncheck:** "Disable auto-create new users" (to allow SSO to create accounts).
- **Check:** "Allow users to connect social logins with their account".
- **Check:** "Update user profile every login".

3.2 Custom OpenID Connect

Click the + button under Custom OpenID Connect and fill in the following:

Nextcloud Field	Authentik URL Path
Internal name	authentik
Authorize URL	https://auth.goonersnas.com/application/o/authorize/
Token URL	https://auth.goonersnas.com/application/o/token/
User info URL	https://auth.goonersnas.com/application/o/userinfo/
Scope	openid profile email

> **Important:** Ensure there are no leading or trailing spaces in the Client ID or Client Secret fields.

Step 4: Final Testing

1. Open an **Incognito Window**.
2. Navigate to your Nextcloud URL.
3. Click the large **Authentik** button at the bottom of the login form.
4. Log in with your Authentik credentials.

> **Success!** If redirected to your Nextcloud dashboard, the SSO link is active. New users created in Authentik will now automatically have Nextcloud accounts provisioned on their first login.

Wazuh

Installing Wazuh Central Components Natively on Ubuntu Server

This guide provides a streamlined procedure for installing the Wazuh indexer, Wazuh server, and Wazuh dashboard on a single Ubuntu node. This native installation avoids the complexities of container networking and volume permissions.

Prerequisites

- A fresh Ubuntu Server installation (Recommended: 22.04 or 24.04 LTS).
- Root or **sudo** privileges.
- Minimum Hardware: 8 vCPU, 16 GB RAM (Wazuh is resource-intensive).

Step 1: System Preparation

Before starting the installation, ensure the OS is up to date and the identity of the server is established.

1. Set the Hostname

Identify the server clearly on your network (e.g., SecuServer).

```
sudo hostnamectl set-hostname SecuServer
```

2. Update System Packages

Ensure all base repositories and installed packages are current.

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

3. Reboot

Apply any kernel updates or hostname changes.

```
sudo reboot
```

Step 2: The Wazuh Installation

Wazuh provides an installation assistant that automates certificate generation and component linking. We use a version-specific URL to ensure the script downloads correctly.

1. Download and Run the Assistant

The **-a** flag performs an automated all-in-one installation.

```
curl -sO https://packages.wazuh.com/4.14/wazuh-install.sh && sudo bash ./wazuh-install.sh -a
```

2. Wait for Completion

The script will install the Indexer, Server, and Dashboard. This typically takes 5–15 minutes depending on hardware and network speed.

Step 3: Securing Credentials

Once the installation finishes, the terminal will display a **Summary** block.

> **CRITICAL:** Copy the admin password immediately. It is randomly generated and required for the first login.

If you lose this password, you can retrieve it from the installation files by running:

```
sudo tar -O -xvf wazuh-install-files.tar wazuh-install-files/wazuh-passwords.txt
```

Step 4: Accessing the Dashboard

Wazuh natively listens on port **443** and enforces **HTTPS**.

1. Open your browser and navigate to: **https://<your_server_ip>**
2. **Bypass SSL Warning:** Since the certificates are self-signed by the script, your browser will show a "Not Secure" warning. Click **Advanced** and then **Proceed**.
3. **Login:** Use the username **admin** and the password saved from Step 3.

Troubleshooting Note: Port 80 vs. 443

If the page refuses to connect, ensure you are explicitly typing **https://** in the address bar. Wazuh does not listen on standard HTTP (Port 80) and will not automatically redirect you.

Managing Users and Roles in Wazuh (v4.14+)

Wazuh operates on a **two-layered security model**. Because it is built on top of OpenSearch, user management is split into two parts:

1. **OpenSearch (Indexer):** Controls access to the underlying database and system settings.
2. **Wazuh App:** Controls access to security agents, rules, and dashboards.

To give a user full administrative privileges, you must grant them permissions in *both* layers. This guide outlines how to properly create an admin, how to assign restricted roles, and what those roles mean.

Part 1: How to Create a Super Admin User

Creating a Super Admin requires setting up the user in the Indexer, assigning them a specific "Backend role" to bypass system locks, and then mapping them to the Wazuh App.

Step 1: Create the User & Assign the Backend Role (Database Layer)

The **all_access** system role is locked by default. To make someone an admin, you must assign them the **admin** Backend Role, which automatically inherits full system access.

1. Click the **Global Menu (≡)** in the top left corner.
2. Navigate to **Indexer management** -> **Security**.
3. Click on **Internal users**, then click the blue **Create internal user** button.
4. Enter a descriptive **Username** and a secure **Password**.
5. Scroll down to the **Backend roles** section. Type `admin` into the box and click **Add another backend role** (or press Enter).
6. Click the blue **Create** (or Save) button at the bottom right.

Step 2: Map the User to the Wazuh App (Application Layer)

Now that the user has database access, you must grant them control over the Wazuh security features.

1. Click the **W. logo** in the top left to return to the Wazuh App.
2. Navigate to **Server management** -> **Security**.
3. Click the **Roles mapping** tab at the top of the screen.
4. Click the blue **Create Role mapping** button.
5. **Role mapping name:** Give it a recognizable name (e.g., *username_admin_access*).
6. **Roles:** Check the box for `administrator`.
7. Scroll down to the **Mapping rules** section. Under **Map internal users**, click the dropdown and select the user you created in Step 1.
8. *Crucial:* If there is a default "Custom rule" (e.g., one that looks for the word "wazuh"), click the **red trash can icon** to delete it.
9. Click **Save role mapping**.

The user is now a full Super Admin and can log in with total system control.

Part 2: How to Assign Standard Roles (e.g., Read-Only)

If you want to create a restricted user (like a junior analyst who can only view alerts but cannot change settings), the process is nearly identical, but you skip the `admin` backend role.

1. **Create the User:** Go to *Indexer management* -> *Security* -> *Internal users* and create the user. **Do not** add anything to the "Backend roles" section. Just save the username and password.
2. **Map the Role:** Go to *Server management* -> *Security* -> *Roles mapping* and click **Create Role mapping**.
3. **Select the Restricted Role:** In the Roles dropdown, select a restricted role like `readonly` instead of administrator.
4. **Map the User:** Select your new user from the "Map internal users" dropdown, delete any default custom rules, and save.

Understanding Built-In Roles

Wazuh comes with several pre-configured roles that dictate what a user can see and do within the application.

- **administrator:** Full control. The user can deploy agents, edit configuration files, create custom security rules, and manage other users.
- **readonly:** View-only access. The user can look at security events, read dashboard metrics, and check agent status, but cannot modify rules or system settings.
- **agents_admin:** Agent management only. The user can deploy, group, and manage endpoints, but cannot change core server configurations or indexer settings.
- **agents_readonly:** Granular view access. The user can only view data and alerts coming from the agents, without access to administrative menus.

> **Note:** Always ensure that users requiring global server configuration access are granted the `admin` Backend Role in the Indexer management menu, as Wazuh App roles alone cannot override database-level restrictions.

Installing Wazuh Agent on Linux (Ubuntu/Debian)

Overview

This guide outlines the procedure for installing the Wazuh agent on Debian-based systems. It includes the standard repository method for local LAN devices, as well as a direct package installation method for devices on restricted networks (e.g., forced OpenVPN tunnels) where standard DNS or APT updates fail.

Prerequisites

- Root or **sudo** privileges on the target Linux machine.
- Connectivity to the Wazuh Manager.
- Port **1514/tcp** (data) and **1515/tcp** (enrollment) open on the Manager's firewall.

Method 1: Standard APT Installation (Main Network)

Use this method for standard servers that have unrestricted outbound internet access to resolve and update package lists.

1. Repository Configuration

Import the Wazuh GPG key and add the official repository to your package manager's sources.

```
curl -s https://packages.wazuh.com/key/GPG-KEY-WAZUH | sudo gpg --no-default-keyring --keyring g

echo "deb [signed-by=/usr/share/keyrings/wazuh.gpg] https://packages.wazuh.com/4.x/apt/ stable m

sudo apt-get update
```

2. Agent Installation

Install the agent while passing the Manager's FQDN as an environment variable.

```
sudo WAZUH_MANAGER='wazuh.goonersnas.com' apt-get install wazuh-agent
```

Method 2: Direct Package Installation (VPN/Restricted Networks)

Use this method if the host is behind a strict VPN that blocks or fails to resolve standard APT update servers. This bypasses the package manager entirely.

1. Download the Package

Fetch the specific .deb installer directly from the Wazuh servers.

```
wget https://packages.wazuh.com/4.x/apt/pool/main/w/wazuh-agent/wazuh-agent_4.14.5-1_amd64.deb
```

2. Force Installation via DPKG

Install the downloaded package, passing the local IP address instead of the domain name to bypass local DNS resolution failures.

```
sudo WAZUH_MANAGER='192.168.0.153' dpkg -i wazuh-agent_4.14.5-1_amd64.deb
```

Service Activation & Verification

Once the agent is installed, enable it to start on boot and initiate the service.

```
sudo systemctl daemon-reload
sudo systemctl enable wazuh-agent
sudo systemctl start wazuh-agent
```

Troubleshooting Common Issues

1. Error: "Job for wazuh-agent.service failed because a timeout was exceeded"

On newer Linux kernels or slower hardware, the systemd default timeout is often too short for the Wazuh initialization process. Extend the timeout with an override file:

```
sudo mkdir -p /etc/systemd/system/wazuh-agent.service.d/
echo -e "[Service]\nTimeoutStartSec=300" | sudo tee /etc/systemd/system/wazuh-agent.service.d/ti
sudo systemctl daemon-reload
sudo systemctl restart wazuh-agent
```

2. Manual Enrollment (If client.keys is empty)

If the agent installs but fails to retrieve a key from the manager, trigger enrollment manually:

```
sudo /var/ossec/bin/agent-auth -m 192.168.0.153
```

3. Fix: Agent Stuck in "Pending" or "Never Connected" Status

If the agent shows as registered in the dashboard but is "never connected", or if checking the local state (`sudo grep ^status /var/ossec/var/run/wazuh-agentd.state`) shows `status='pending'`, the agent cannot resolve or reach the Manager's address. Update the configuration to use the static IP.

Step A: Edit the configuration file

```
sudo nano /var/ossec/etc/ossec.conf
```

Step B: Locate the <client> block and update the <address> to the Manager's IP

```
<client>
  <server>
    <address>192.168.0.153</address>
    <port>1514</port>
    <protocol>tcp</protocol>
  </server>
</client>
```

Step C: Restart the agent to apply changes

```
sudo systemctl restart wazuh-agent
```

Peanut

Peanut

How to Connect PeaNUT (CasaOS) to a Synology UPS

Monitoring your power infrastructure is an excellent way to protect your data. If your uninterruptible power supply (UPS) is already connected to your Synology NAS via USB, you can use the NAS as a Network UPS Tools (NUT) server. This setup allows the PeaNUT container running on your CasaOS machine to pull live metrics like battery level, current load, and remaining runtime.

Step 1: Configure the Synology NAS (The Server)

First, we need to configure the Synology NAS to broadcast its UPS data on your local network so that CasaOS can read it.

1. Log in to your **Synology DSM**.
2. Navigate to **Control Panel** > **Hardware & Power** > **UPS**.
3. Ensure **Enable UPS support** is checked (it should be if the UPS is already recognized).
4. Check the box for **Enable network UPS server**.
5. Click the **Permitted DiskStation Devices** button.
6. In the pop-up window, enter the **IP Address of your CasaOS machine**. This step specifically grants PeaNUT permission to access the UPS data.
7. Click **OK**, and then click **Apply** to save the settings.

Step 2: Connect PeaNUT in CasaOS (The Client)

Synology uses a strict, hardcoded set of default credentials for its internal NUT server. When you open your PeaNUT dashboard to add the UPS connection, you must use these exact details.

```
Host / IP Address: [The IP Address of your Synology NAS]
Port: 3493
UPS Name: ups
Username: monuser
```

```
Password: secret
```

> **Note:** `monuser` and `secret` are the permanent default credentials Synology uses for its network UPS server. The UPS name is also defaulted to `ups`.

Once you enter those details and hit save, PeaNUT will connect to the Synology NAS and begin populating your dashboard with live data.

Matomo

Setting Up Matomo Analytics with a Dedicated MariaDB Instance

Overview

This article details the process of deploying Matomo Analytics using a dedicated, isolated MariaDB database. This method ensures that your analytics data remains separate from other services (like Nextcloud), preventing database conflicts and simplifying maintenance.

Prerequisites

- **Host System:** Ubuntu Server with CasaOS or Docker installed.
- **Storage Path:** /DATA/AppData/matomo
- **Port Availability:** Port 8484

1. Docker Compose Configuration

Use the following configuration for a Custom Install in CasaOS:

```
services:
  matomo_db:
    image: mariadb:10.11
    container_name: matomo_db
    command: --max-allowed-packet=64MB
    environment:
      - MARIADB_ROOT_PASSWORD=matomo_root_password
      - MARIADB_DATABASE=matomo
      - MARIADB_USER=matomo
      - MARIADB_PASSWORD=matomo_secure
    volumes:
      - /DATA/AppData/matomo/db:/var/lib/mysql
    restart: unless-stopped

  matomo_app:
    image: matomo:latest
    container_name: matomo_app
    restart: unless-stopped
```

```
    ports:
      - "8484:80"
    environment:
      - MATOMO_DATABASE_HOST=matomo_db
      - MATOMO_DATABASE_DBNAME=matomo
      - MATOMO_DATABASE_USERNAME=matomo
      - MATOMO_DATABASE_PASSWORD=matomo_secure
    volumes:
      - /DATA/AppData/matomo/html:/var/www/html
    depends_on:
      - matomo_db
```

2. Database Connection Credentials

During the Matomo web-based setup wizard, use these internal credentials:

Setting	Value
Database Server	matomo_db
Login	matomo
Password	matomo_secure
Database Name	matomo

3. Implementation on BookStack

1. Log in to Matomo and navigate to **Administration > Websites > Manage**.
2. Copy the **JavaScript Tracking Code**.
3. In BookStack, navigate to **Settings > Customization**.
4. Paste the script into the **Custom HTML Head Content** area and save.

Integrating Matomo Analytics with BookStack

Overview

Once Matomo is installed, you must link it to BookStack to begin capturing visitor data, page views, and search trends. This is done by injecting a small JavaScript snippet into the BookStack header.

1. Retrieve Tracking Code from Matomo

1. Log in to your **Matomo** dashboard (e.g., http://192.168.0.152:8484).
2. Click the **Gear Icon** (Administration) in the top right.
3. On the left sidebar, navigate to **Websites > Tracking Code**.
4. Ensure the correct website (e.g., "BookStack") is selected in the dropdown.
5. In the **JavaScript Tracking Code** section, copy the entire block of code.

2. Apply Code to BookStack

1. Open your **BookStack** instance and log in as an **Administrator**.
2. Go to **Settings** (Gear icon in top navigation).
3. Select **Customization** from the left-hand menu.
4. Find the section labeled **Custom HTML Head Content**.
5. Paste your copied Matomo script directly into the text area.
6. Scroll to the bottom and click **Save Settings**.

```
<!-- Matomo Tracking Script Example -->
<script>
  var _paq = window._paq = window._paq || [];
  _paq.push(['trackPageView']);
  _paq.push(['enableLinkTracking']);
  (function() {
    var u="//YOUR_SERVER_IP:8484/";
    _paq.push(['setTrackerUrl', u+'matomo.php']);
    _paq.push(['setSiteId', '1']);
    var d=document, g=d.createElement('script'), s=d.getElementsByTagName('script')[0];
    g.async=true; g.src=u+'matomo.js'; s.parentNode.insertBefore(g,s);
  })();
</script>
```

3. Verification

- Open your BookStack homepage in a new tab.
- Navigate to the **Dashboard** in Matomo.
- Check the **Visits in Real-time** widget; your activity should appear within seconds.

Note: If you use an ad-blocker, you may need to disable it for your local domain to see your own tracking data.

Matomo

Integrating Nextcloud Hub 28 with Matomo Analytics

This article outlines the steps to connect your self-hosted Nextcloud instance (v33.0.0+) to Matomo for tracking user activity and file browsing.

---

1. Prerequisites

- **Matomo Site ID:** Ensure you have created a "Measurable" for Nextcloud. For this setup, the **Site ID is 3**.
- **Nextcloud Version:** Hub 28 Winter (v33.0.0).
- **App Version:** Matomo Tracking (v1.0.0).

---

2. Matomo Configuration

1. Log in to your Matomo instance at `https://matomo.goonersnas.com/`.
2. Navigate to **Administration (Cog Icon) > Websites > Manage**.
3. Verify that the URL for Site ID 3 is set to `https://nc.goonersnas.com/`.

---

3. Nextcloud App Installation

1. Log in to Nextcloud as an **Administrator**.
2. Go to **Apps** and select the **Integration** category.
3. Search for **"Matomo Tracking"**.
   - *Note: Avoid the legacy "Piwik" version (v0.13.0) as it is incompatible with v33.*
4. Click **Download and enable** (you may need to click "Allow untested app").

---

4. Connection Settings

1. Navigate to **Administration settings > Additional settings**.

2. Locate the **Matomo Tracking** section.
3. Enter the following details:

```
Matomo Server Url: https://matomo.goonersnas.com/ Site ID: 3 [x] Track file
browsing [x] Track user id
```

Note: Ensure the URL includes the trailing slash. Settings save automatically upon losing focus from the text fields.

---

5. Verification

1. Open Nextcloud in an Incognito/Private browser window and navigate through a few folders.
2. Return to Matomo and go to **Dashboard > Real-time > Visitor Log**.
3. Look for active visits originating from your Nextcloud domain.

---

6. Troubleshooting & Security

Content Security Policy (CSP)

If tracking fails to report, you may need to whitelist your Matomo domain in the Nextcloud `config.php` file:

```
'custom_csp_policy' => "default-src 'self'; script-src 'self'
https://matomo.goonersnas.com; img-src 'self' data:
https://matomo.goonersnas.com; connect-src 'self'
https://matomo.goonersnas.com;",
```

Nginx Proxy Manager

Ensure that your Matomo proxy host does not have "Block Common Exploits" enabled if it interferes with the tracking `.js` delivery to external subdomains.

HestiaCP

NordVPN

NordVPN Installation and Strict Configuration on Lubuntu

This guide provides a step-by-step process for installing NordVPN on a Lubuntu system and configuring a secure "Panama-only" connection with a kill switch while maintaining local network accessibility for Remote Desktop (XRDP) or NAS access.

1. Installation and Permissions

First, install the NordVPN repository and the application. Then, ensure your user has the necessary permissions to control the service.

```
# Download and install the NordVPN repo
sh <(curl -sSf https://downloads.nordcdn.com/apps/linux/install.sh)

# Add your user to the nordvpn group
sudo usermod -aG nordvpn $USER

# Refresh group permissions (Requires your Lubuntu password)
su - $USER
```

2. Authentication and Whitelisting

Authentication is a two-part process when working over SSH or XRDP. You must whitelist local subnets **before** enabling the kill switch to avoid a lockout.

```
# 1. Run login to generate your unique URL
nordvpn login

# 2. Copy the URL to a browser, log in, and copy the resulting callback URL
# 3. Paste your unique callback URL into the command below
nordvpn login --callback "nordvpn://login?action=login&exchange_token..."
```

```
# 4. Whitelist local subnets for Remote Desktop/NAS access
nordvpn whitelist add subnet 192.168.0.0/24
nordvpn whitelist add subnet 192.168.100.0/24
```

3. Configuring the "Panama-Only" Lock

Configure the client to automatically connect to Panama and kill the internet connection if the VPN tunnel drops.

```
# Set autoconnect to Panama
nordvpn set autoconnect on Panama

# Connect to the VPN
nordvpn connect Panama

# Enable the Kill Switch
nordvpn set killswitch on
```

4. Troubleshooting: "nordvpn.service is masked"

If the service reports as "masked" or fails to start, it is likely due to legacy SysV script conflicts or broken package dependencies (like synosnap) blocking the daemon. Use this "Nuclear Reset" sequence.

```
# 1. Remove legacy SysV init conflict
sudo rm -f /etc/init.d/nordvpn

# 2. Nuke phantom mask files across all system paths
sudo rm -f /etc/systemd/system/nordvpn.service /run/systemd/system/nordvpn.service
/lib/systemd/system/nordvpn.service /usr/lib/systemd/system/nordvpn.service

# 3. Purge blocking agents and fix broken package states
sudo dpkg --purge synosnap
sudo apt install -f
```

```
# 4. Reinstall and force-unmask the daemon
sudo apt install --reinstall nordvpn -y
sudo systemctl daemon-reload
sudo systemctl unmask nordvpn
sudo systemctl enable --now nordvpn
```

5. Management and Verification

Commands for daily use and verification.

```
# Check current connection status
nordvpn status

# View all current settings
nordvpn settings

# Check if the background service is running
sudo systemctl status nordvpn
```