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.