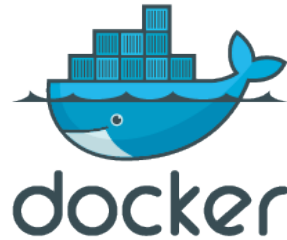
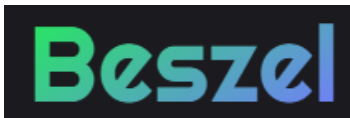


Quickstart Guide



--



Introduction

Architecture

Beszel consists of two main components:

- **Hub:** A web application that provides a dashboard for viewing and managing connected systems. Built on [PocketBase](#).
- **Agent:** Runs on each system you want to monitor, creating a minimal SSH server to communicate system metrics to the hub.

Step 1: Create the Necessary Folders

If you have multiple nodes and a Docker Swarm environment, you can reference the [GlusterFS guide](#) for distributing folders across nodes. The Folders needs to be Accasable from every Node and be updated in realtime.

```
mkdir -p /mnt/glustermount/data/beszel_data
```

Step 2: Docker Compose or Portainer for Initial Setup

You can either create a `docker-compose.yml` file manually or use Portainer to set up Beszel. Below is an example configuration that should work out of the box for a standalone Docker setup.

IMPORTANT: After you add a new system in the Beszel web UI, you must update the `KEY` value with your public key (provided by the hub) and then restart the agent service. Also, use `host.docker.internal` as the Host/IP when prompted, instead of `localhost` or `127.0.0.1`.

Always make sure your beszel-hub starts first. So on the Agent you add the Option:
"`depends_on:`"

Docker Standalone Example

```
services:
  beszel:
    image: henrygd/beszel:latest
    container_name: beszel
    restart: unless-stopped
    extra_hosts:
      - host.docker.internal:host-gateway
    ports:
      - 8090:8090
    volumes:
      - /mnt/glustermount/data/beszel_data:/beszel_data

  beszel-agent:
    image: henrygd/beszel-agent:latest
    container_name: beszel-agent
    restart: unless-stopped
    network_mode: host
    volumes:
```

```
- /var/run/docker.sock:/var/run/docker.sock:ro
environment:
  PORT: 45876
  # Do not remove quotes around the key
  KEY: 'UPDATE WITH YOUR PUBLIC KEY (copy from "Add system" dialog)'
```

Docker Swarm + Traefik Example

```
version: "3.7"

services:
  beszel:
    image: henrygd/beszel:latest
    container_name: beszel
    restart: unless-stopped
    networks:
      - management_net
    extra_hosts:
      - host.docker.internal:host-gateway
    ports:
      - 8090:8090
    volumes:
      - /mnt/glustermount/data/beszel_data:/beszel_data
    deploy:
      mode: replicated
      replicas: 1
      labels:
        - "traefik.enable=true"
        - "traefik.http.services.beszel-agent.loadbalancer.server.port=8090"

  beszel-agent:
    image: henrygd/beszel-agent:latest
    container_name: beszel-agent
    restart: unless-stopped
    network_mode: host
    depends_on:
      - beszel_beszel
    ports:
      - 45876:45876
    volumes:
```

```
- /var/run/docker.sock:/var/run/docker.sock:ro
environment:
  PORT: 45876
  # Do not remove quotes around the key
  KEY: 'UPDATE WITH YOUR PUBLIC KEY (copy from "Add system" dialog)'
deploy:
  mode: replicated
  replicas: 1
  labels:
    - "traefik.enable=true"
    - "traefik.http.services.beszel-agent.loadbalancer.server.port=45876"

networks:
  management_net:
    external: true
```

In Docker Swarm the Name-Conventions might be a little different, thats why the Depends_on Option uses `beszel_beszel` and not only `beszel`. because the first `beszel` stands for the stackname and the second `beszel` for the service name defined in the docker-compose

Why `network_mode: host` ?

The agent must use host network mode to access network interface metrics, which automatically exposes the port. If you do not need network statistics, you can remove `network_mode: host` and map the port manually in the Compose file.

Step 3: Start the Containers

Once you've created your Compose file, you can deploy the services. The process differs slightly depending on your setup:

- **Docker Swarm:**

```
docker stack deploy -c docker-compose.yaml beszel
```

- **Docker Standalone:**

```
docker compose up -d
```

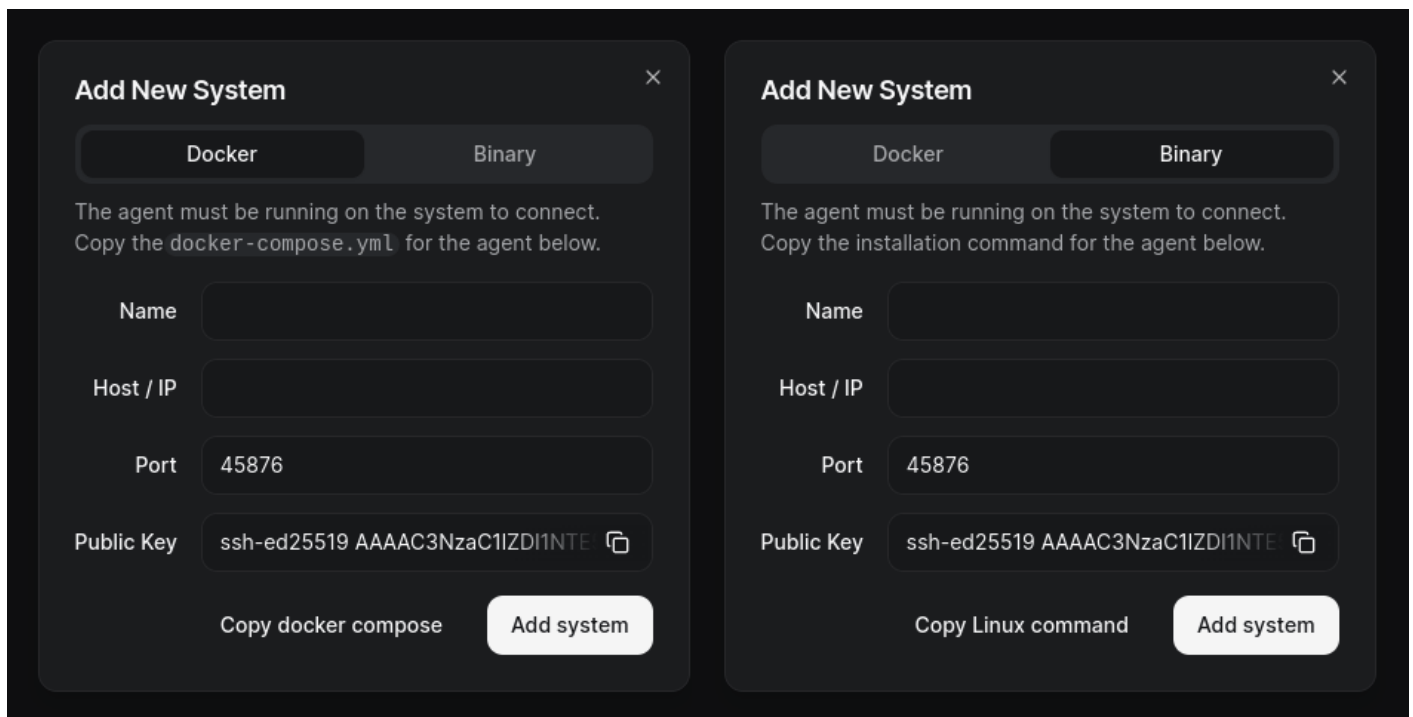
- **Portainer:** Use the Portainer UI to import the `docker-compose.yaml` file and start the stack.

Step 4: Create an Admin User

Open `http://localhost:8090` (or your chosen URL/port) in your browser and follow the prompts to create an admin user.

Step 5: Adding Systems / Nodes

When you add a new system from the Beszel hub's web UI, you'll be given a snippet for the `beszel-agent` configuration (Docker Compose or a binary install command). Copy the public key from the "Add system" dialog and place it in the `KEY` variable of your agent container or service.



The image shows two side-by-side screenshots of the "Add New System" dialog in the Beszel web UI. Both dialogs have a close button (X) in the top right corner. The left dialog has "Docker" selected as the installation method, and the right dialog has "Binary" selected. Both dialogs contain the following fields: "Name" (text input), "Host / IP" (text input), "Port" (text input with "45876" entered), and "Public Key" (text input with "ssh-ed25519 AAAAC3NzaC1lZDI1NTE..." entered and a copy icon). Below the fields, the left dialog has a "Copy docker compose" button and an "Add system" button. The right dialog has a "Copy Linux command" button and an "Add system" button. The text above the fields in both dialogs states: "The agent must be running on the system to connect. Copy the [docker-compose.yml / installation command] for the agent below."

Different Agents on Different Nodes:

If you plan to monitor multiple nodes with Docker Swarm, you can create separate agent services, each running on a different node with constraints to ensure proper placement.

Make sure to give each Agent an unique Servicename.

Make sure to give each Agent an unique Port and also change that Port in the Web-UI when adding a new System.

```
beszel-agent1:
  image: henrygd/beszel-agent:latest
  restart: unless-stopped
```

```
network_mode: host
depends_on:
  - beszel_beszel
volumes:
  - /var/run/docker.sock:/var/run/docker.sock:ro
ports:
  - 45876:45876
environment:
  PORT: 45876
  KEY: 'YOUR_PUBLIC_KEY_FROM_HUB'
deploy:
  mode: replicated
  replicas: 1
  labels:
    - "traefik.enable=true"
    - "traefik.http.services.beszel-agent1.loadbalancer.server.port=45876"
  placement:
    constraints:
      - node.hostname == swarmpi1 # Only deploy on node 'swarmpi1'
```

```
beszel-agent2:
  image: henrygd/beszel-agent:latest
  restart: unless-stopped
  network_mode: host
  depends_on:
    - beszel_beszel
  volumes:
    - /var/run/docker.sock:/var/run/docker.sock:ro
  ports:
    - 45877:45877
  environment:
    PORT: 45877
    KEY: 'YOUR_PUBLIC_KEY_FROM_HUB'
  deploy:
    mode: replicated
    replicas: 1
    labels:
      - "traefik.enable=true"
      - "traefik.http.services.beszel-agent2.loadbalancer.server.port=45877"
  placement:
```

constraints:

- node.hostname == swarmpi2 # Only deploy on node 'swarmpi2'

beszel-agent3:

image: henrygd/beszel-agent:latest

restart: unless-stopped

network_mode: host

depends_on:

- beszel_beszel

volumes:

- /var/run/docker.sock:/var/run/docker.sock:ro

ports:

- 45878:45878

environment:

PORT: 45878

KEY: 'YOUR_PUBLIC_KEY_FROM_HUB'

deploy:

mode: replicated

replicas: 1

labels:

- "traefik.enable=true"
- "traefik.http.services.beszel-agent2.loadbalancer.server.port=45878"

placement:

constraints:

- node.hostname == swarmpi3 # Only deploy on node 'swarmpi3'

Each agent service can then monitor the node on which it is running, allowing you to collect and consolidate metrics across your entire cluster.

Troubleshooting

If you have Problems with this Quicksetup Guide or its just not running, go visit my [Troubleshooting-Section](#).

Revision #5

Created 1 February 2025 22:51:16 by aeoneros

Updated 4 February 2025 08:47:53 by aeoneros