> Hinweis: Nur für Multiplikatoren

JupyterHub in systemweiter venv mit Mehrbenutzerbetrieb, Admin-Zugriff und systemweiten MQTT- und GSON-Bibliotheken

Diese Anleitung ist eine **Blaupause** für die Bereitstellung eines zentralen **JupyterLab-Servers für Lerngruppen**. Sie beschreibt die grundsätzliche Einrichtung von JupyterHub für mehrere Benutzer (PAM-Authentifizierung) und definiert den Lehrenden (`pdal`) als **Administrator**. Diese Admin-Rolle ist entscheidend für den Support, da sie den Zugriff auf die Sessions der Lernenden ermöglicht. Zusätzlich werden systemweite Kernel (Java, R) und IoT-Bibliotheken (Paho-MQTT, GSON) für alle Nutzer bereitgestellt.

---

1. Voraussetzungen installieren

Erstellen sie einen Standard-LXC mit Ubuntu 24.4. Wechseln sie zu User `pdal`, führen Update und Upgrade aus.

```
sudo apt install -y python3 python3-venv python3-pip nodejs npm default-jdk git
curl unzip r-base
```

Erklärung: Installiert die notwendigen Grundpakete für Python (JupyterHub), Node.js (Proxy), Java (Java-Kernel), R (R-Kernel), Git (für Repos), curl/unzip (für Download und Entpacken). Man kann die einzelnen Pakete auch der Reihe nach installieren und muss diese nicht in eine Befehlszeile packen.

2. Zentrales Bibliotheksverzeichnis erstellen

```
sudo mkdir -p /opt/jupyterhub-libs/{python,java,r}
sudo chmod -R a+rx /opt/jupyterhub-libs
```

Erklärung: Erstellt ein zentrales Verzeichnis, das von allen Benutzern genutzt werden kann – z. B. für gemeinsame Bibliotheken, Beispielskripte oder Wrapper.

3. Systemweite venv für JupyterHub einrichten

```
sudo mkdir -p /opt/jupyterhub-venv
sudo chown root:root /opt/jupyterhub-venv
sudo python3 -m venv /opt/jupyterhub-venv
```

Befehl	Erklärung

Befehl	Erklärung
`sudo mkdir -p /opt/jupyterhub-venv`	Erstellt das Zielverzeichnis für die virtuelle Python-Umgebung unter `/opt`. Die Option `-p` sorgt dafür, dass übergeordnete Verzeichnisse ebenfalls angelegt werden, falls sie noch nicht existieren.
`sudo chown root:root /opt/jupyterhub-venv`	Setzt den Besitzer des Verzeichnisses auf `root`, damit nur administrative Nutzer Änderungen daran vornehmen können.
`sudo python3 -m venv /opt/jupyterhub-venv`	Erstellt die virtuelle Umgebung (venv) im angegebenen Verzeichnis. Diese isolierte Python-Umgebung erlaubt die Installation von JupyterHub und dessen Abhängigkeiten, ohne das globale System-Python zu verändern.

Erklärung: Erzeugt eine isolierte Python-Umgebung, in der alle JupyterHub-Komponenten unabhängig vom System python verwaltet werden.

4. JupyterHub und Lab installieren

```
source /opt/jupyterhub-venv/bin/activate
sudo /opt/jupyterhub-venv/bin/python -m pip install --upgrade pip
sudo /opt/jupyterhub-venv/bin/python -m pip install jupyterhub notebook jupyterlab
deactivate
sudo npm install -g configurable-http-proxy
```

Erklärung: Installiert alle Jupyter-Komponenten innerhalb der `venv` sowie den HTTP-Proxy (erforderlich für JupyterHub). JupyterLab wird mit installiert.

5. JupyterHub-Konfiguration erzeugen

```
sudo mkdir -p /etc/jupyterhub
cd /etc/jupyterhub
sudo /opt/jupyterhub-venv/bin/jupyterhub --generate-config
```

Erklärung: Erzeugt die Standardkonfigurationsdatei `jupyterhub\_config.py`, die später angepasst wird, um Nutzer, Spawner und Admin-Zugriff zu definieren.

6. Konfiguration anpassen (inkl. Admin-Benutzer pdal)

Bearbeite `/etc/jupyterhub/jupyterhub_config.py`:

```
sudo nano /etc/jupyterhub/jupyterhub_config.py
```

```
c = get_config()

# Authentifizierung über Systembenutzer
c.JupyterHub.authenticator_class = 'jupyterhub.auth.PAMAuthenticator'
c.Authenticator.allow_existing_users = True
# c.Authenticator.allow_all = True

# Start direkt in JupyterLab
c.Spawner.default_url = '/lab'
# Funktion zum erstellen des Verzeichnis 'notebooks'
import pwd
import os


def create_notebook_dir(spawner):
    username = spawner.user.name
    notebook_dir = f"/home/{username}/notebooks"
    os.makedirs(notebook_dir, exist_ok=True)
    uid = pwd.getpwnam(username).pw_uid
    gid = pwd.getpwnam(username).pw_gid
    os.chown(notebook_dir, uid, gid)
    # Wichtig: `Spawner.notebook_dir` dynamisch setzen
    spawner.notebook_dir = notebook_dir

c.Spawner.pre_spawn_hook = create_notebook_dir
# Admin-Rechte für Benutzer 'pdal'
c.Authenticator.admin_users = {'pdal'}
c.JupyterHub.admin_access = True

# Zusätzliche Umgebungsvariablen für Bibliothekspfad
import os
c.Spawner.environment = {
    'CLASSPATH': '/opt/jupyterhub-libs/java/*',
    'R_LIBS_USER': '/opt/jupyterhub-libs/r',
    'PYTHONPATH': '/opt/jupyterhub-libs/python'
}
```

Erklärung:

- Nutzt Systembenutzer (PAM) zur Authentifizierung – es sind keine separaten JupyterHub-Zugänge nötig.
- Startet direkt in JupyterLab (`/lab`) statt im klassischen Jupyter-Notebook-Interface.
- Beim Starten der Sitzung wird automatisch geprüft, ob ein persönliches Verzeichnis `~/notebooks` existiert – falls nicht, wird es automatisch für den jeweiligen Benutzer angelegt und korrekt berechtigt.
- Die Konfiguration `pre_spawn_hook` setzt den Pfad zu den Notebooks dynamisch pro Nutzer, sodass jeder standardmäßig im Verzeichnis `/home/username/notebooks` landet.
- Der Benutzer `pdal` erhält Adminrechte – er darf u. a. auf Sessions anderer Benutzer zugreifen (z. B. zur Unterstützung).

- Setzt zentrale Umgebungsvariablen für:
    - Java-Bibliotheken im Verzeichnis `/opt/jupyterhub-libs/java/*` (`CLASSPATH`)
    - R-Bibliotheken im Verzeichnis `/opt/jupyterhub-libs/r` (`R_LIBS_USER`), die systemweit von allen Usern genutzt werden können.

7. Systembenutzer anlegen

```
sudo adduser alice
sudo adduser bob
# pdal ist bereits vorhanden
```

Erklärung: Erstellt Benutzer, die sich per PAM bei JupyterHub anmelden können. Ohne Systembenutzer ist keine Anmeldung möglich.

> Hinweis: Bevor sich die erstellten Systemuser am Jupyterhub anmelden können, muss der Admin diese User zuvor noch im Adminpanel in Jupyterhub hinzufügen, da Sie sonst keine Berechtigung haben auf den Jupyterhub zu zugreifen

8. systemd-Dienst für JupyterHub einrichten

Datei `/etc/systemd/system/jupyterhub.service` mit `sudo nano /etc/systemd/system/jupyterhub.service` erstellen und bearbeiten:

```
[Unit]
Description=JupyterHub (venv)
After=network.target

[Service]
User=root
WorkingDirectory=/etc/jupyterhub
Environment="PATH=/opt/jupyterhub-
venv/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin"
Environment="CLASSPATH=/opt/jupyterhub-libs/java/*"
Environment="R_LIBS_USER=/opt/jupyterhub-libs/r"
Environment="PYTHONPATH=/opt/jupyterhub-libs/python"
ExecStart=/opt/jupyterhub-venv/bin/jupyterhub
Restart=on-failure

[Install]
WantedBy=multi-user.target
```

Abschnitt	Direktive	Beschreibung
[Unit]	Description	Kurze Beschreibung des Dienstes.

Abschnitt	Direktive	Beschreibung
[Unit]	After	Bestimmt die Abhängigkeit, hier soll der Dienst erst nach dem Netzwerk starten.
[Service]	User	Benutzer, unter dem der Dienst läuft (hier `root`, kann aber auch ein anderer Nutzer sein).
[Service]	WorkingDirectory	Arbeitsverzeichnis, von dem aus der Dienst startet (hier der Konfigurationsordner).
[Service]	Environment	Setzt Umgebungsvariablen, z. B. PATH, CLASSPATH, R_LIBS_USER für den Dienstprozess.
[Service]	ExecStart	Kommando, das zum Starten des Dienstes ausgeführt wird (hier JupyterHub in der venv).
[Service]	Restart	Verhalten bei Absturz oder Fehler — hier Neustart bei Fehlern.
[Install]	WantedBy	Bestimmt, wann der Dienst gestartet wird (hier bei Multi-User-Runlevel, also normalem Boot).

```
sudo systemctl daemon-reexec
sudo systemctl enable --now jupyterhub
```

Erklärung: Erstellt einen Dienst, der JupyterHub automatisch beim Booten startet und bei Fehlern neu lädt. Wichtig für Dauerbetrieb.

9. JupyterHub im Browser aufrufen

```
http://<server-ip>:8000
```

Mit User `pdal` anmelden. Unter Menü-Punkt "File" -> "Hub Control Panel" -> "Admin" -> "add Users" die Nutzer "bob" und "alice" hinzufügen. Erst danach können sich diese User anmelden. Für weitere User erst einen User im System anlegen (`adduser xyz`) und dann im "Hub Control Panel" hinzufügen.

Als Admin können sie sich auf laufende Sessions von anderen Usern einloggen, Server stoppen oder starten. So können sie die Lernenden supporten.

10. IJava-Kernel (Java) installieren

```
cd /opt
sudo wget https://github.com/SpencerPark/IJava/releases/download/v1.3.0/ijava-1.3.0.zip
sudo unzip ijava-1.3.0.zip
```

Um den Kernel Systemweit installieren zu können benötigen wir noch das Paket Jupyter-client, welches wir uns mit

```
sudo apt install python3-jupyter-client
```

installieren.

Jetzt können wir die folgenden Befehle ausführen

```
sudo python3 install.py --sys-prefix
```

Erklärung: Installiert einen Java-Kernel für Jupyter systemweit. Notebooks mit Java-Code sind nun möglich, z. B. für Softwareentwicklung oder Informatikunterricht.

11. IRKernel (R) installieren

> Hinweis: Installieren den R-Kernel nur wenn sie ihn wirklich brauchen; ist groß und dauer lange.

Zunächst in die R-Console wechsel:

```
sudo R
```

In der R-Console ausführen:

```
install.packages("IRkernel")
Sys.setenv(PATH = paste(Sys.getenv("PATH"), "/opt/jupyterhub-venv/bin", sep =
":"))
IRkernel::installspec(user = FALSE)
q()
```

Danach die R-Console mit `q()` wieder verlassen.

Erklärung: Fügt systemweit einen R-Kernel hinzu – nutzbar für Statistik, Data Science oder Forschung. Es wird sichergestellt, daß temporär der Jupyter-Befehl auch in der Umgebung von R verfügbar ist.

12. Paho-MQTT für Python installieren

Wechseln sie zu dem `root`-User. Starten sie das virtuelle Envoirement mit :

```
source /opt/jupyterhub-venv/bin/activate
```

Dann die Bibliotheken installieren - vergewissern sie sich das das "venv" läuft (`(jupyterhub-venv) root@jupyterhub:`).

```
pip install paho-mqtt
deactivate
```

Erklärung: Installiert die MQTT-Bibliothek für Python und legt Beispielcode zentral ab. Alle Nutzer können damit MQTT-Daten verarbeiten, z. B. in IoT-Projekten.

13. Paho-MQTT für Java + GSON hinzufügen

```
cd /opt/jupyterhub-libs/java
# MQTT
sudo curl -LO https://repo.eclipse.org/content/repositories/paho-
releases/org/eclipse/paho/org.eclipse.paho.client.mqttv3/1.2.5/org.eclipse.paho.cl
ient.mqttv3-1.2.5.jar
# GSON
sudo curl -LO
https://repo1.maven.org/maven2/com/google/code/gson/gson/2.10.1/gson-2.10.1.jar
```

Damit ist PahoMQTT und GSON systemweit verfügbar. Die Bibliotheken können in den Jupyter-Notebooks (Java) wie folgt verwendet werden:

```
// 1. MQTT-Bibliothek laden
%classpath add jar /opt/jupyterhub-libs/java/org.eclipse.paho.client.mqttv3-
1.2.5.jar

// 2. GSON-Bibliothek laden (falls diese auch verwendet wird)
%classpath add jar /opt/jupyterhub-libs/java/gson-2.10.1.jar
import org.eclipse.paho.client.mqttv3.*;
import com.google.gson.*;
```

Erklärung: Java-MQTT-Client sowie die GSON-Bibliothek (für JSON-Parsing) werden als `.jar` zentral abgelegt. So können alle Java-Notebooks beide Bibliotheken direkt verwenden.

14. Zugriffsrechte auf zentrale Bibliotheken

```
sudo chmod -R a+rx /opt/jupyterhub-libs
```

Erklärung: Alle Benutzer erhalten Lese- und Ausführungsrechte auf zentrale Ressourcen – z. B. Beispielnotebooks oder gemeinsam genutzte Bibliotheken.

Verzeichnisübersicht

Pfad	Inhalt
`/opt/jupyterhub-venv`	Python-Umgebung mit JupyterHub
`/etc/jupyterhub`	Konfiguration von JupyterHub
`/opt/IJava`	Java-Kernel (IJava)
`/opt/jupyterhub-libs/python/`	Gemeinsame Python-Skripte und Beispielcode
`/opt/jupyterhub-libs/java/`	Paho MQTT + GSON Java-Bibliotheken (.jar)
`/opt/jupyterhub-libs/r/`	Gemeinsame R-Skripte (MQTT/Wrapper/Beispiele)
`/usr/local/share/jupyter/`	Systemweit registrierte Jupyter-Kernel

Hinweis zur Sicherheit

Für produktive Nutzung: Eine Absicherung per HTTPS oder ein Reverse-Proxy (z. B. mit NGINX oder Apache2) wird dringend empfohlen. So werden Passwörter und Notebook-Inhalte nicht im Klartext übertragen. Die Konfiguration zu SSL Verschlüsselung kann in der offiziellen Dokumentation von Jupyterhub nachgelesen werden. https://jupyterhub.readthedocs.io/en/stable/tutorial/getting-started/security-basics.html

Das Einbinden der Zertifikate beschreibt eine andere Dokumentation die hier zu finden ist: 2050 CA-sslmitSANZertifikat.

Quellen

- „Documentation · IRkernel“. Zugegriffen: 28. Juli 2025. [Online]. Verfügbar unter: https://irkernel.github.io/docs/
- „Installation Basics“, JupyterHub. Zugegriffen: 31. Juli 2025. [Online]. Verfügbar unter: https://jupyterhub.readthedocs.io/en/stable/tutorial/installation-basics.html
- „Security settings“, JupyterHub. Zugegriffen: 31. Juli 2025. [Online]. Verfügbar unter: https://jupyterhub.readthedocs.io/en/stable/tutorial/getting-started/security-basics.html

---

Lizenz