Infrastructure as Code

Dieses Repository beinhaltet die Ansible-Playbooks und Konfigurationsdateien zur Verwaltung meines Arch Linux Laptops (Hyprland) sowie meiner Server-Infrastruktur. Ziel ist es, die Einrichtung neuer Maschinen zu automatisieren und Konfigurationen konsistent zu halten.

Struktur des Repositories

- **setup.yml**: Haupt-Playbook für den lokalen Laptop (Arch Linux). Installiert Pakete und verteilt Dotfiles.
- **server_setup.yml**: Playbook für die Server-Verwaltung (Docker, Security, Updates).
- **inventory**: Inventar-Datei für den lokalen Host (Laptop).
- **inventory_server**: Inventar-Datei für Remote-Server (Hetzner, Home-Server).
- **files/**: Enthält die originalen Konfigurationsdateien (Hyprland, Waybar, Skripte etc.), die von Ansible verteilt werden.

Anleitung: Laptop (Neuinstallation)

Voraussetzung: Eine minimale Arch Linux Installation mit funktionierendem Internetzugang.

1. Vorbereitung

Zunächst müssen Git und Ansible manuell installiert werden, um das Repository zu klonen.

```
sudo pacman -S git ansible
```

2. Repository klonen

Das Repository wird in das Home-Verzeichnis geklont.

```
mkdir -p ~/git
cd ~/git
git clone https://git.deinedomain.de/DeinUser/arch-setup.git
cd arch-setup
```

3. Installation ausführen

Das Playbook `setup.yml` nutzt die lokale Verbindung. Der Parameter `-K` ist notwendig, um das `sudo`-Passwort für die Paketinstallation abzufragen.

```
ansible-playbook setup.yml -K
```

Hinweis: Wenn das Playbook durchgelaufen ist, sollte ein Neustart erfolgen, damit alle Dienste (Hyprland, Login-Manager) korrekt greifen.

Anleitung: Server Management

Dieses Playbook dient dazu, Server (Hetzner VPS, Home-Server) zu aktualisieren oder neu aufzusetzen.

Voraussetzungen

- Der SSH Public Key des Laptops muss auf den Servern in `~/.ssh/authorized_keys` hinterlegt sein.
- Die IP-Adressen müssen in der Datei `inventory_server` aktuell sein.

Ausführung

Vom Laptop aus wird folgendes Kommando ausgeführt:

```
ansible-playbook -i inventory_server server_setup.yml -K
```

Dies führt Updates durch, prüft Docker-Container und stellt sicher, dass grundlegende Tools installiert sind.

Workflow für Änderungen

Um die Konfigurationen sauber zu halten, sollte nicht direkt in `~/.config` gearbeitet werden, sondern immer über das Repository.

1. Änderungen an Dateien im Ordner `files/` vornehmen oder neue Pakete in `setup.yml` eintragen.

2. Ansible Playbook lokal ausführen (`ansible-playbook setup.yml -K`), um Änderungen zu testen und anzuwenden.
3. Wenn alles funktioniert: Commit und Push zum Git-Server.

```
git add .
git commit -m "Beschreibung der Änderung"
git push
```

Troubleshooting

Fehler: "Connection refused" auf localhost

Tritt auf, wenn Ansible versucht, sich per SSH mit dem eigenen Laptop zu verbinden, anstatt lokal zu arbeiten.

Lösung: Sicherstellen, dass in der Datei `setup.yml` die Zeile `connection: local` enthalten ist oder dass in der Datei `inventory` folgender Eintrag steht: `localhost ansible_connection=local`

Fehler: "Permission denied (publickey)" bei Servern

Ansible kann sich nicht mit dem Remote-Server verbinden.

Lösung:

1. Prüfen, ob der SSH-Agent läuft und der Key geladen ist (`ssh-add -l`).
2. Manuelle Verbindung testen: `ssh user@server-ip`.
3. Falls die IP neu ist, muss sie ggf. erst in die `known_hosts` aufgenommen werden (einmal manuell per SSH verbinden und bestätigen).

Fehler: Ansible überschreibt Änderungen immer wieder

Wenn Ansible bei jedem Lauf "changed" anzeigt, obwohl nichts geändert wurde.

Lösung: Oft stimmen die Dateirechte nicht überein. Im Playbook sicherstellen, dass `mode: preserve` genutzt wird, oder explizite Rechte (`0644` für Dateien, `0755` für Ordner) gesetzt sind, die mit dem Ist-Zustand übereinstimmen.

Git Push funktioniert nicht (Authentifizierung)

Falls nach dem Passwort gefragt wird, aber der Login fehlschlägt.

Lösung:

1. Prüfen, ob die Remote-URL korrekt ist (`git remote -v`).
2. Sicherstellen, dass Port 2222 (oder der konfigurierte Git-Port) genutzt wird, falls SSH verwendet wird.
3. Bei HTTPS: Prüfen, ob das Passwort korrekt ist. Ggf. `git config --global credential.helper store` nutzen, um das Passwort einmalig dauerhaft zu speichern.

Revision #2
Created 2025-12-30 12:42:59 UTC by Denode
Updated 2025-12-30 12:43:35 UTC by Denode